André Marques Cavalcanti

DESENVOLVIMENTO E ANÁLISE DE ALGORITMOS PROBABILÍSTICOS DE OTIMIZAÇÃO GLOBAL

Recife

2004

Universidade Federal de Pernambuco
Programa de Pós-graduação em Engenharia Elétrica

DESENVOLVIMENTO E ANÁLISE DE ALGORITMOS PROBABILÍSTICOS DE OTIMIZAÇÃO GLOBAL

Tese

submetida à Universidade Federal de Pernambuco

como parte dos requisitos para obtenção do grau de

Doutor em Engenharia Elétrica

André Marques Cavalcanti

Recife, Julho de 2004.

DESENVOLVIMENTO E ANÁLISE DE ALGORITMOS PROBABILÍSTICOS DE OTIMIZAÇÃO GLOBAL

André Marques Cavalcanti

A comissão examinadora composta pelos professores: FERNANDO MENEZES CAMPELLO DE SOUZA, DES/UFPE, GERALDO LEITE TORRES, DEESP/UFPE, MANOEL AFONSO DE CARAVALHO JÚNIOR, DEESP/UFPE, ADIEL TEIXIERA DE ALMEIDA, DEP/UFPE e FRANCISCO DE SOUSA RAMOS, DCE/UFPE sob a presidência do Prof. Joaquim Ferreira Martins Filho, Coordenador do Programa de Pós-Graduação em Engenharia Elétrica, consideram o candidato **ANDRÉ MARQUES CAVALCANTI APROVADO.**

"Tudo na natureza opera de acordo com leis. Só os seres racionais têm a faculdade de agir segundo a concepção de leis, ou seja, de princípios, o que equivale a dizer que eles têm uma vontade. Visto que deduzir ações exige razão, a vontade nada mais é que a razão prática. Se a razão determina infalivelmente a vontade, as ações de um ser racional, reconhecidas como necessárias de um ponto de vista objetivo, também o são ao nível do subconsciente; o que é o mesmo que dizer que a vontade é a faculdade de escolher somente o que a razão, independentemente da inclinação, reconhece como necessário praticamente, ou seja, como bom. Mas, se por si mesma a razão não determina suficientemente a vontade, se esta última se submete igualmente a condições subconscientes (a impulsos particulares) que não coincidam com as condições objetivas, em uma palavra, se a vontade não está em si mesma em perfeita concordância com a razão (o que é na realidade o caso para os homens), então as ações que são reconhecidas objetivamente como necessárias são fortuitas para o subconsciente, e a determinação dessa vontade segundo as leis objetivas é uma obrigação; o que significa que a relação entre as leis objetivas e uma vontade que não é inteiramente boa é concebida como a determinação da vontade de um ser racional por princípios de razão, mas que a vontade, a natureza, não segue necessariamente"

IMANNUEL KANT(1724-1804)

Dedico este trabalho à aqueles que fazem parte da minha razão. Em especial ao meu pai Antonio (In memoriam), a minha mãe Isis, a minha esposa Alice e aos meus filhos: Kátia, Theresa e André.

Agradecimentos

No momento em que finalizamos um trabalho não podemos deixar de fazer uma reflexão. Analisar o quanto ficamos ausentes dos nossos familiares, amigos e digamos um pouco dispersos em nossas obrigações de rotina. Refletir sobre a importância dos agentes que nos reanimaram em momentos de dúvida e cansaço, e sobre aqueles que nos emprestaram o seu conhecimento e experiência. São eles os verdadeiros construtores deste projeto que não se reflete só neste trabalho e sim no nosso comprometimento, na nossa forma de observar e interagir com o mundo a partir de então. A transformação pela qual passamos torna-se possível de entender quando paramos para pensar na contribuição mais que profissional, de amigo, de colega e de um grande mestre, aquele que considero Mega Mestre o Prof. Fernando Menezes Campello de Souza, o meu orientador.

Quero destacar a presença do meu anjo da guarda na Secretaria do Curso Andréa Tenório, que me ajudou muito nas questões administrativas. Perpetuar fatos importantes ocorridos durante este trabalho um deles foi a descoberta de que mais solitário que poderia parecer não estava só, ao olhar para os lados verifiquei a presença de pessoas que se transformaram sileciosamente em amigos e de amigos mais amigos. A cada um o meu muito obrigado por tê-los em minha vida com a certeza de que farei tudo para que assim permaneçam.

André Marques Cavalcanti

Universidade Federal de Pernambuco

30 de Julho de 2004

Resumo da Tese apresentada à UFPE como parte dos requisitos necessários para

obtenção do grau de Doutor em Engenharia Elétrica.

DESENVOLVIMENTO E ANÁLISE DE ALGORITMOS PROBABILÍSTICOS DE OTIMIZAÇÃO GLOBAL

André Marques Cavalcanti

Julho/2004

Orientador: Fernando Menezes Campello de Souza, PhD
Área de Concentração: Processamento de Energia
Palavras-chaves: otimização global, algoritmos probabilísticos, heurísticas
Número de páginas: xiii+164

O desenvolvimento de algoritmos de otimização global irrestrita tem sido pesquisado na tentativa de obter algoritmos gerais que apresentem um bom desempenho em classes abrangentes de problemas de otimização. Os métodos que utilizam derivadas apresentam um bom desempenho mas, na grande maioria dos problemas esta informação não está disponível ou é de grande dificuldade a sua obtenção. Existe uma grande dificuldade de se estabelecer algoritmos eficientes sem a informação da derivada. Os métodos de busca direta se apoiam em heurísticas para determinar a direção de busca com certa eficiência, porém sem garantia de convergência. Este trabalho realiza uma avaliação do problema da convergência dos algoritmos de busca direta, desenvolve um modelo geral, estratégias de abordagem e melhorias, analisa e implementa um algoritmo que é capaz de determinar a direção de busca do ótimo de forma eficiente, sem o uso de derivadas. O *software* desenvolvido utiliza nuvens probabilísticas que fornecem o conhecimento da performance da distribuição de probabilidade da nuvem de pontos. Essa distribuição através dos seus momentos fornece a direção de aproximação do ponto de ótimo. Na presente pesquisa o algoritmo melhorado proposto apresentou uma excelente performance em relação às listas de *Benchmark* existentes e aceitas pela comunidade de otimização global. O aspecto mais relevante é a abrangência de classes de problemas possíveis de resolver com este algoritmo sem perda de performance em reconhecer a classe de problemas a ele submetido.

Abstract of Thesis presented to UFPE as a partial fulfillment of the requirements for the degree of Doctor in Electrical Engineering.

DEVELOPMENT AND ANALISYS OF PROBABILISTIC GLOBAL OPTIMIZATION ALGORITHMS

André Marques Cavalcanti

July/2004

Supervisor: Fernando Menezes Campello de Souza, PhD
Area of Concentration: Power Processement
Keywords: global optimization, probabilistic algorithms, heurisctics
Number of pages: xiii+164

The unconstrained global optimization development has been researched in an attempt to obtain high performance general algorithms relevant to a wide range of optimization problems. Derivative methods have good performance but are difficult to apply in many problems. It is very difficult to establish efficient algorithms that do not use derivatives. Heuristic based direct search methods can determine the search direction with some efficiency, however there are no convergence guarantees. This study evaluates the direct search algorithm convergence problem, develops a general model, establishes search convergence strategies, determine improvements, analysis and implements an algorithm capable of efficiently determining the optimum search direction without the use of derivatives. The software developed uses probabilistic cloud distribution performance knowledge. Utilizing momentum, this probabilistic distribution provides the optimum point approach direction. This research indicates that the improved algorithm presented an excellent performance in relation to community approved global Benchmarks. A relevant aspect in this study is the wide class of problems this algorithm can be applied to without performance loss.

Conteúdo

Lista de Tabelas

Lista de Figuras

Capítulo 1

INTRODUÇÃO

O homem sempre desejou entender o ambiente onde vive, o seu mundo. Na medida das suas necessidades, o processo de compreensão deste ambiente foi se tornando mais complexo. Já na antiguidade, inúmeras e sofisticadas estruturas de abstração foram se apresentando para representar as propriedades e os diversos graus de interação entre os vários interferentes desse todo. Na impossibilidade de lidar diretamente com a complexidade do mundo, o homem tem desenvolvido a capacidade de representá-lo através de metáforas para representar e resolver a sua relação com este mundo, [3].

O processo da busca da interpretação bem estruturada da realidade é fundamentalmente um processo de modelagem.

A definição do modelo representa a interpretação aproximada de um fenômeno real. Nesta perspectiva os modelos são representações simplificadas da realidade que devem preservar, para determinadas situações e enfoque, uma equivalência adequada. O poder de representatividade é a característica que o torna desejável. A capacidade de simplificação lhe confere a capacidade de reproduzir os apectos operacionais dos fatos observados.

Em [4] afirma-se que para obter modelos eficientes são necessários pelo menos três habilidades:

1. Foco Holístico: Avalia se a solução impacta significativamente sobre outros contextos e questiona se a solução cria outros problemas que posteriormente possam vir a anular a combinação apresentada.

2. Tratamento Eclético da Dimensão de Análise: Os métodos de solução a serem utilizados devem ser o mais livremente dispostos. Epistemologia e Axiologia não devem ser consideradas como bases dictômicas da modelagem , mas complementares.

3. Tradução Adequada: Uma boa tradução contextual pode ser expressa através de um correto isomorfismo entre o fenômeno e seu modelo.

O processo de tradução contextual deve ser capaz de identificar os elementos fundamentais da questão da transformação para uma representação capaz de ser manipulada por artifícios ou métodos de solução. Na medida em que a tradução produz uma representação mais ou menos tratável pelos métodos existentes, a utilizibilidade é assim definida.

O conceito que representa a interferência da tradução na possibilidade de tradução é denominado de complexidade.

Define-se **complexidade** por suas propriedades que ocasionam e interferem no fenômeno.

- A primeira propriedade a destacar é a **permeabilidade** ao meio ambiente circunvizinho. Um modelo simples possui um perímetro de interferências simples e bem definido.

- A segunda propriedade está associada à **morfologia**. Um modelo simples possui uma estrutura homogênea, morfologia uniforme e um número reduzido de variáveis.

- A última propriedade engloba a **dinâmica**, correspondendo a considerações de como a estrutura interna se altera ao longo do tempo.

Um modelo não é igual à realidade, mas suficientemente similar para que as conclusões obtidas através de sua análise ou operação possam ser estendidas à realidade. Em conseqüência, para a formalização do modelo é indispensável definir:

1. A estrutura relacional entre os subsistemas representados.

2. O comportamento funcional de cada subsistema ou comportamento atômico.

3. Os fluxos de inter-relacionamentos.

Da própria definição de modelo e de seus objetivos derivam as principais características dos modelos de otimização.

1. As propriedades analíticas.

2. Melhoria mensurável do processo.

3. Reconhecimento das interações do modelo e sobre o modelo.

Nesta perspectiva busca-se estabelecer métodos que permitam o estabelecimento de modelos que melhor representem os problemas reais sem se preocupar com as limitações teóricas para a resolução.

1.1 O Problema de Otimização Global

Em engenharia, economia, administração, etc., decisões quantitativas são freqüentemente modeladas por ferramentas e conceitos de otimização. A Teoria da Decisão tipicamente deseja encontrar a decisão ótima absoluta que corresponde ao mínimo (ou máximo) de uma certa função objetivo, satisfazendo a um dado conjunto de possíveis restrições. A função objetivo retrata a performance total do sistema, tais como: risco, perda, utilidade, benefício ou erro. As restrições são de natureza fisíca, técnica, econômica, dentre outras.

As técnicas de otimização usadas apresentam um grande número de problemas de natureza aplicada que não são resolvidos satisfatoriamente por métodos de programação linear. Uma vasta gama de problemas de natureza aplicada pertence à classe de funções não convexas, inviabilizando o uso de técnicas tradicionais.

Os problemas da classe dos não lineares complexos estão associados a modelos de tomadas de decisão. Apresentam com freqüência múltiplos ótimos locais. Em muitos casos reais o número de soluções locais não é conhecido *a priori*, e a qualidade da solução global e local difere substancialmente. Portanto, esses modelos são de difícil solução, e as regras e estratégias de solução padrão não podem ser aplicadas diretamente para resolvê-los. Então, é necessário utilizar técnicas e conceitos de otimização global (OG).

Uma importante motivação para o estudo de modelos de otimização global é dada pela necessidade do uso adequado de modelos de descrição e análise da maioria dos problemas.

Otimização global é uma das grandes áreas de questionamentos e interesses que, em princípio, cobre toda a programação matemática tradicional (PM) e controle ótimo

(CO). Isto implica que otimização global necessita estender paradigmas e estratégias de soluções, em adição e combinação das técnicas e conceitos clássicos.

O objetivo primordial da tomada de decisão empresarial é a maximização da utilidade do decisor. Na prática nem sempre é traduzida pela maximização do lucro da empresa ou minimização dos custos de produção (dos objetivos declarados ou não), por vezes carregada de um viés subjetivo do decisor. A programação matemática apoia a tomada de decisão. De uma forma geral, pode-se dizer que a contribuição da Otimização (ou Programação Matemática) dentro da modelagem e solução de problemas de decisão em casos reais é:

- Estabelecer melhorias mensuráveis na operação do sistema, automatizar processos e identificar gargalos operacionais.

- Desenvolver análises comparativas de desempenho operacional, determinar valores de referência para os diversos produtos em diferentes estágios da cadeia de produção, processamento, estocagem e transporte.

Assumindo que em um dado problema de otimização interessa obter a solução global. Para se obter a solução ótima necessita-se de uma condição suficiente para um ponto do conjunto viável ser um mínimo (ou máximo), aplicando-se a expressão "minimização global" às questões relativas ao seguinte problema:

- Encontrar o menor valor de f sobre todo C, onde f é uma função real definida num conjunto C.

- Encontrar o mínimo global de f sobre C , isto é, um $x^* \in C$ para o qual $f(x^*) \leq f(x)$. Sendo $f(x)$ a função objetivo definida como $f : \mathbb{R}^n \to \mathbb{R}$ sobre um conjunto C.

1.1.1 A Problemática

A solução dos problemas de otimização utilizando métodos determinísticos exige que as condições de otimalidade sejam satisfeitas. Estes algoritmos reconhecem pontos como soluções verificando se satisfazem às várias condições de otimalidade.

As restrições impostas pelos métodos determinísticos têm estimulado o desenvolvimento dos métodos probabilísticos que não fazem nenhuma imposição à função objetivo, desde que esta seja mensurável. Estes métodos exploram a capacidade de processamento dos computadores, uma vez que, basicamente fazem uma busca exploratória em todo o intervalo viável sem impor nenhuma condição à função objetivo. A dificuldade do uso destes métodos encontra-se na determinação da direção a ser tomada, no espaço de busca, para encontrar a solução ótima. Desta forma, a pesquisa está direcionada para as técnicas de busca e nas regras para definição do tamanho do passo.

A motivação da pesquisa em métodos probabilísticos advém da necessidade de algoritmos eficientes na solução de classes mais amplas de problemas de OG, e, em particular, probabilísticos, pela possibilidade de resolução em estruturas computacionais paralelas. Isto também ocorre em uma análise comparativa com os algoritmos de busca indireta (os determinísticos) que são considerados de precisão infinita, devido à condição necessária para que x^* seja um ponto de ótimo exato de f ser $\nabla f(x^*) = 0$, no caso determinístico. Porém, em um algoritmo iterativo, pelo fato dos cálculos serem realizados com precisão finita, esta condição passa a ser $\nabla f(x^*) \approx 0$ e desta forma o critério de convergência do algoritmo depende de uma aproximação do ótimo considerada aceitável, tal como nos modelos de busca direta.

1.2 O Estado da Arte

Otimização Global é a caracterização do mínimo (máximo) global de uma função não linear. Os problemas de Otimização Global estão espalhados em uma faixa muito grande de aplicações de modelagem matemática para sistemas reais, tais como:

- Engenharia elétrica:
  - alocação de usinas termoelétricas para suprimento da demanda em função da redução da capacidade de produção hidroelétrica devido ao período de seca,
  - controle de entrada e saída de unidades de produção de energia complementar (energia eólica),
- Engenharia de telecomunicações:
  - aplicações em problemas de codificação,
  - otimização de tráfego na rede [5],
  - gerenciamento de catástrofe [6];
- Mercado e economia:
  - previsão de vendas [7] e [8],
  - comportamento de mercados de capitiais;
- Engenharia de produção:
  - determinação ótima de ciclos de manutenção [9];
- Informática:

- inteligência artificial,
- processamento paralelo.

Quando se trata da solução de problemas de programação não linear aceita-se tipicamente o ótimo local como solução, pelo fato de ser freqüentemente inviável a obtenção do ótimo global. Conforme [10] existe uma vasta literatura sobre o assunto, embora não exista algoritmo eficiente para a solução global. A afirmação da inexistência de algoritmo eficiente está baseada no fato de que a busca pelo ótimo depende fortemente da capacidade de processamento do computador utilizado e não do algoritmo propriamente dito.

Métodos específicos de otimização têm sido desenvolvidos para classes específicas de problemas de otimização. Adicionalmente, técnicas gerais têm sido pesquisadas para estender a uma vasta gama de classes de problemas [10], [11], [12], [13], [14], [15].

- **Problema Combinatorial**. Tem uma função linear ou não-linear definida sobre um conjunto de soluções finito, porém muito grande. Existe um número significativo de categorias de problemas de otimização combinatorial, incluíndo problemas de redes, cronograma e transporte. Se a função é parcialmente linear, o problema combinatorial pode ser resolvido com uma combinação de programação inteira e *Branch and Bound*. Métodos heurísticos tipo *Simulated Annealing*, Busca Tabu e Genético também são aplicados para encontrar a solução aproximada.

- **Problema geral sem restrições**. Tem uma função não-linear com domínio no espaço euclidiano $\mathbb{R}^n$. Uma variedade de estratégias de particionamento tem

sido proposta para resolver este problema. Tipicamente estes métodos baseiam-se no conhecimento *a priori* de como rapidamente a função pode variar ou da avaliação da formulação analítica e topológica da função objetivo para determinar intervalos razoáveis de busca (métodos intervalares). Métodos estatísticos também usam o particionamento para decompor o espaço viável, porém ele usa o conhecimento *a priori* de como a função objetivo pode ser remodelada. Uma grande variedade de outros métodos tem sido proposta para resolver este problema de forma aproximada, incluindo *Simulated Annealing*, Genético, *Clustering*, *Path Ant* e método continuado.

- **Problema geral com restrições** Tem uma função não-linear no espaço euclidiano $\mathbb{R}^n$ de restrições. Este tipo de problema não tem recebido muita atenção como o problema geral irrestrito em soluções usando os métodos probabilísticos. Entretanto, muitos dos métodos usados para o modelo irrestrito têm sido adaptados para este problema.

Devido à dificuldade da obtenção de métodos gerais de solução dos problemas de OG é vasta a literatura sobre o assunto. Na atualidade buscam-se modelos baseados em: derivadas (da classe determinística), probabilísticos (*drive search*), heurísticas (*path ant, simulated annealing*) e híbridos.

Os algoritmos que se apoiam em modelos estocásticos estão sendo vistos com um maior interesse no desenvolvimento de algoritmos inteligentes. Estes algoritmos são capazes de orientar a busca pela solução ótima de forma eficiente mesmo quando as funções objetivo têm topologia complexa. São utilizados métodos de busca do ótimo através de processos evolutivos na escolha dos elementos do espaço viável.

As técnicas que se utilizam de probabilidade para estimar a solução de problemas de certa complexidade datam do século XVIII, com a agulha de Buffon - Georges Louis Leclerc (1707 -1788), algoritmo proposto para estimar o valor de π [2].

Em 1953, N. Metropolis *et al* [16] propõem o algoritmo *Simulated Annealing* inspirado em fenômenos físicos da solidificação e cristalização de diversos materiais.

Em 1975, John Holland propõe o algoritmo Genético baseado em processos naturais evolutivos de Darwin (1858).

Em 1986, o Prof. Fernando Menezes Campello de Souza propõe o desenvolvimento de um algoritmo baseado na transformação da função objetivo em uma função de distribuição de probabilidade. Neste sentido destacam-se a Dissertação de mestrado [17] que implementa este algoritmo, denominado de TALUS, em uma planilha realizando diversos testes com funções reconhecidas pela comunidade de OG como apropriadas para teste. Em 1999 são desenvolvidas melhorias no TALUS, trabalho apresentado como dissertação de mestrado em ngenharia elétrica, quanto a função de estanciamento por [2] e implementação do novo algoritmo em linguagem C dedicado a testes das funções escolhidas anteriormente para avaliação de performance. Ainda em 1999 [18] em sua dissertação de mestrado desenvolve a implementação eletrônica do TALUS.

O natural crescimento tecnológico e econômico tem impulsionando o desenvolvimento de algoritmos para resolver problemas complexos de otimização. Portanto, a busca de algoritmos que atendam uma maior classe de problemas é altamente relevante, pois, evitaria a necessidade de simulações com diversos algoritmos. Além da certeza de que no momento que for introduzida ou retirada alguma restrição ou variável não será necessário reavaliar se o método é adequado ou não.

Pesquisas recentes, desenvolvidas por Campello de Souza [2], [18], [17] e [19], comprovam que o algoritmo probabilístico TALUS apresenta uma alta performance na solução de problemas clássicos de otimização propostos em [20] e [21]. Uma abordagem bastante abrangente é apresentada nos trabalhos acima necessitando de uma ampla divulgação e desenvolvimentos ainda quanto ao *software* e discussão analítica do processo interno do algoritmo.

1.3 Objetivos e Organização

Através da análise de recentes pesquisas em algoritmos para solução de problemas não-lineares de otimização, identifica-se a inexistência de métodos de aplicação geral. A proposta da utilização de algoritmos probabilísticos tem dado uma nova perspectiva ao desenvolvimento da solução a problemas de otimização com função objetivo não-convexa e não contínua. Dentre os algoritmos probabilísticos *Simulated Annealing*, Genético, TALUS, entre outros o TALUS tem apresentado resultados satisfatórios. Neste sentido pretende-se descrever analiticamente os critérios de convergência, além de desenvolver o algoritmo para permitir a sua aplicação com qualquer função objetivo a ser otimizada. Propõe-se também realizar alguns ajustes internos para obter uma melhor performance.

1.3.1 Objetivos Gerais

1. Estabelecer a metodologia geral para desenvolvimento de algoritmos de busca direta.

2. Estabelecer um critério geral de convergência dos algoritmos de busca direta.

3. Demonstrar analiticamente a convergência do algoritmo TALUS quando satisfeita a hipótese da existência de um ótimo global no espaço de busca.

4. Elaborar e implementar um *software* para o TALUS.

1.3.2 Objetivos Específicos

1. Estabelecer os valores ótimos dos parâmetros internos do algoritmo TALUS.

2. Estabelecer critérios de convergência para algoritmos de busca direta.

1.3.3 Organização

No **Capítulo 2** esboça-se os princípios teóricos gerais da otimização e os modelos não-lineares com restrição e sem restrição são sumarizados. Uma ênfase especial é dada aos **métodos irrestritos** de Otimização Global, considerado neste trabalho por estar associado à aplicação de procedimentos na solução de problemas de otimização global.

No **Capítulo 3** uma pesquisa na literatura sobre algoritmos de Otimização Global é apresentada. Métodos como *Path Ant* , Algoritmo Genético e *Simulated Annealing* são apresentados. Os problemas de suas aplicações são sumarizados. As carências correntes e as potenciais contribuições para métodos de OG são expostas.

As contribuições teóricas deste trabalho são apresentadas nos **Capítulos 4, 5 e 6**.

O **Capítulo 4** são apresentadas as hipóteses para a construção de modelos de busca direta e as estruturas dos principais algoritmos de OG juntamente com os resultados analíticos da convergência e outras melhorias.

No **Capítulo 5** são apresentados os pressupostos básicos para a construção do algoritmo TALUS, a **base teórica da direção da busca eficiente** para os casos unidimensional e n-dimensional e fundamentos funcionais do TALUS. Os **resultados analíticos da convergência** são mostrados a partir da análise dos seus aspectos funcionais.

No **Capítulo 6** são discutidos os aspectos funcionais dos parâmetros internos do TALUS. O objetivo é identificar quais critérios devem ser adotados quando da avaliação da eficiência, observando o tempo de processamento e a precisão do resultado. São realizados vários testes de desempenho utilizando parte do conjunto de funções de teste de Moré. Estas funções são tidas como *benchmarking* pela comunidade científica de otimização. Além disto é realizada uma **análise comparativa** com outros algoritmos (ASA e Geno3).

Os resultados e conclusões são sumarizados no **Capítulo 7**, onde também, tópicos para trabalhos futuros são indicados.

Capítulo 2

ASPECTOS TEÓRICOS DA OTIMIZAÇÃO

Os problemas de otimização podem ser divididos em dois grandes grupos: os convexos e os não-convexos.

Existe um grande número de modelos no domínio do escopo de otimização local tradicional, notadamente o de programação linear e o de programação convexa. Entretanto, existem também numerosos casos nas quais as condições (de linearidade e convexidade) estruturais não são satisfeitas ou não são de fácil verificação. O fenômeno e o processo modelados podem ser não-linear de ordem elevada. Eigen e Winkler [22] apresentam um vasto número de aplicações em diferentes contextos como aqui citados.

Como foi visto no Capítulo 1 os problemas de OG estão associados a problemas de decisão. Neste sentido o desenvolvimento de algoritmos de OG são suportados por vários aspectos matemáticos fundamentais não só por questões teóricas, importantíssimas, mas também pelas necessidades que se apresentam na elaboração e implemen-

tação de algoritmos. Destacando-se:

- Natureza dos espaços: euclidianos ou não-euclidianos, finitos ou infinitos, contínuos ou discretos, vetoriais ou não-vetoriais, numéricos ou não-numéricos ou de ordem pequena ou grande.

- Natureza das funções objetivo: contínuas ou descontínuas, diferenciáveis ou não-diferenciáveis, integráveis ou não-integráveis, convexas ou não-convexas, bem-comportadas (funções que não apresentam variações bruscas e muitas oscilações no espaço de busca) ou não.

- Natureza das funções que definem as restrições: lineares ou não-lineares, de igualdade ou desiguladade.

- Complexidade analítica das funções quanto ao: número de pontos de sela, número de mínimos e máximos locais e globais.

O objetivo da pesquisa é o desenvolvimento de algoritmos de OG sem restrição, em espaços euclidianos de qualquer dimensão. A hipótese básica assumida é que as funções expressas analiticamente sejam apenas mensuráveis. Quando da consideração da extensão do problema para os casos com restrição, as hipóteses iniciais continuam válidas. A partir destes pressupostos iniciais abrangem-se os casos em que as funções analíticas sejam convexas, ou não-convexas, diferenciáveis ou não-diferenciáveis, de qualquer dimensão e complexidade.

Embora neste capítulo sejam apresentados os aspectos gerais que dão suporte a solução de problemas de otimização global, o foco desta tese são as funções definidas em espaços euclidianos de qualquer dimensão, definidas por expressões analíticas. Tanto

a função objetivo (podendo ser definida por uma expressão analítica, não necessariamente uma função) como as restrições se supõem mensuráveis. Estas funções ainda podem ser convexas ou não, contínuas ou não, com restrição ou sem, de qualquer dimensão.

2.1 Rápida Taxionomia dos Problemas de Otimização Global

Nesta seção busca-se introduzir algumas definições usadas na linguagem de Programação Matemática (PM) que hoje de forma extendida estão associadas aos problemas de Otimização Global OG.

Programação Linear

Programação Linear(PL) é uma ferramenta para resolver problemas de otimização onde a função objetivo e as restrições são lineares. Em 1947, George Dantzig desenvolveu o método Simplex para resolver os problemas de PL. Desde então, PL tem sido usada para resolver problemas em diversas áreas, tais como: da indústria, bancos, educação, petróleo e transporte. Os problemas de PL estão associados a buscar a melhor quantidade para as variáveis envolvidas que atendem às restrições colocadas. Em outras palavras busca-se obter os melhores resultados diante das restrições impostas. No caso de PL a função objetivo e as restrições são lineares.

O algoritmo Simplex

Para os problemas de PL que possuem apenas duas variáveis de decisão é possível resolver por meio de métodos gráficos. Para problemas de PL com muitas variáveis de decisão a alternativa é utilizar o algoritmo Simplex. Para ser possível utilizar este algoritmo, todas as variáveis das restrições devem ser transformadas em não-negativas e as desigualdades em igualdades. Um problema de PL nesta forma é chamado de Problema de Programação Linear padrão (PPL).

O algoritmo Simplex se baseia no fato de que a solução de um PPL encontra-se em um ponto extremo da região viável. A região viável é um conjunto convexo e, portanto, a solução é única.

Programação Inteira

Os problemas de Programação Inteira (PI) são mais difíceis que os de PPL, pois geralmente PI envolve variáveis com valores 0 ou 1. Um dos problemas clássicos associados a PI é o do caixeiro viajante. Este problema se refere a um vendedor que deve visitar várias cidades, porém, deve fazer o precurso mais econômico possível, sem visitar duas vezes a mesma cidade.

Programação Não-Linear

Uma classe de problemas mais abrangentes são os de programação não-linear (PNL). São considerados PNL aqueles cujo função objetivo ou alguma restrição é não-linear.

Uma das diferenças entre os PPL e PNL é que: se a região viável é convexa a solução ótima não necessariamente se encontra em um ponto extremo da região

viável.

Quando da garantia da existência do ótimo uma das condições mais importantes em PNL é a definida por conjuntos convexos ou côncavos, pois, quando a região viável é assim definida, existe a garantia do ótimo global.

Uma das dificuldades em trabalhar com PNL é que nem sempre a região viável é convexa ou côncava. Além disso, muitas vezes a função objetivo não é duas vezes diferenciável. Esta classe de problemas abrange a grande maioria dos problemas de natureza prática nas diversas áreas do conhecimento.

Programação Dinâmica

A Programação Dinâmica (PD) é uma técnica que pode ser usada para resolver muitos problemas de otimização. Em muitas aplicações, PD obtém soluções trabalhando do fim do problema para o começo. Desta maneira, quebra o problema original em uma série de problemas menores, porém mais tratáveis. Primeiro se resolve o primeiro estágio do problema, partindo em seguida para o segundo estágio, continuando subseqüentemente até o último estágio. Em muitas aplicações a decisão é tomada em cada estágio.

Muitas aplicações de PD são reduzidas ao problema de encontrar o menor caminho que une dois pontos em uma rede. Um exemplo de aplicação é utilizado na definição da localização de uma central telefônica o mais próximo possível dos seus usuário e das demais centrais.

Programação Combinatória

Em inúmeras situações, o problema de otimização está associado às alternativas de combinações entre opções, ou entre clientes e suas opções. Identifica-se a necessidade de se estudar a menor distância entre as demandas e os pontos de atendimento. Para esses tipos de problemas, o tomador de decisão necessita ser capaz de exibir uma topologia adequada ao modelo, formas e organizar as configurações desejáveis dentro dessa topologia e critérios de escolha dessas configurações. Nesse tipo de problema a tomada de decisão é polarizada pela arquitetura de ligação, entre três estruturas de representação são extremamanete importantes:

- Caminhos: quando o foco está em uma trajetória;
- Árvores: quando o foco está na continuidade da conexão e no estabelecimento de uma espinha dorsal sobre um conjunto de pontos demandantes;
- Empilhamentos: quando o problema da conexão envolve pequenos grupamentos.

2.2 Os Conjuntos Convexos

Uma das condições usadas na solução de problemas de otimização é a existência de um hiperplano de separação de dois conjuntos convexos.

O teorema de representação dos estados em que, sob condições de restrições regulares, um conjunto convexo pode ser descrito completamente em termos de quantidades, apresentando uma importância geométrica natural, conforme Teorema Fundamental de Inequações Lineares, que tem como conseqüência o Lema de Farkas, [23]. No contexto de programação linear este teorema tem interpretação direta, pois, o ótimo

deve ser obtido no vértice de uma região viável (um ponto extremo de um conjunto convexo), e que o problema pode ter uma solução de fronteira somente se a direção conduzir a uma relação particular com a função objetivo.

Convexidade poliedral ocorre quando existe um número finito de pontos extremos na direção de um ótimo viável.

Tendo em vista a necessidade de estabelecer algumas hipóteses, definições e conceitos para o entendimento dos princípios que permitem ou garantem a existência da OG, alguns deles são apresentados a seguir .

Definição 2.2.1 (Conjunto convexo) *Conforme [24], o conjunto* $S \subseteq \mathbb{R}^p$ *é convexo se* $x, y \in S \Rightarrow \theta x + (1-\theta)y \in S$ *para* $0 \leq \theta \leq 1$. *Equivalentemente,* S *é convexo se todas combinações finitas de pontos em* S *estão em* S, *isto é:*
se $x_i \in S$ *onde* $i = 1, 2...m,\ \sum_{i=1}^{m}\lambda_i = 1$ *para* $\lambda_i \geq 0$ *e* $1 \leq m \leq \infty; \Rightarrow \sum_{i=1}^{m}\lambda_i x_i \in S$

Definição 2.2.2 (Hiperplano) *Em [25], um Hiperplano é um conjunto de pontos* $H(u,v) = \{x : u^T x = v\}$. *Isto mostra que* $H(u,v)$ *separa* $\mathbb{R}^p$ *em dois distintos subespaços:*

$$H^+(u,v) = \{x,\ u^T x > v\}$$

$$H^-(u,v) = \{x,\ u^T x \leq v\}$$

Lema 2.2.1 (Hiperplano de separação - caso simples) *Conforme [25] seja* S *um conjunto convexo fechado em* $\mathbb{R}^p$ *e* x_0 *um ponto não pertencente a* S. *Então existe um hiperplano* H *separando* x_0 *de* S *da seguinte forma* $S \subset H^+,\ x_0 \in H^-$.

Prova: Faça x_1 ser um ponto em S. Então a norma euclidiana do vetor:

$$\inf \|x - x_0\|_2^2 \leq \|x_1 - x_0\|_2^2 = r^2$$

Como a função $\|x-x_0\|_2^2$ é contínua sobre o conjunto fechado $S \cap \{x;\ \|x-x_0\|_2 \leq r\}$, então, o valor mínimo para $x \in S$ é obtido para $x = x^*$. Faz-se $y \in S$ e considera-se $x = x^* + \gamma z$ onde $z = y - x^*$ e $\gamma > 0$, logo:

$$\|x^* + \gamma z - z_0\|_2^2 = \|x^* - x_0\|_2^2 + 2\gamma z^T(x^* - x_0) + \gamma^2\|z\|_2^2$$

Fazendo $\gamma \to 0$, tem-se $z^T(x^* - x_0) \geq 0$. Isto resulta que:

$H(x^* - x_0,\ x^{*T}(x^* - x_0) - \varepsilon)$ é um hiperplano apropriado para todo $\varepsilon > 0$ suficientemente pequeno.

Teorema 2.2.1 *Conforme [25], sejam S e T conjuntos convexos em* $\mathbb{R}^p$, *e* $S \cap T = \varnothing$. *Então existe um hiperplano no qual* $S \subset H^+$, $T \subset H^-$. *Equivalentemente pode-se encontrar* u *tal que:* $\inf_{x \in S} u^T x \geq \sup_{x \in T} u^T x$

Teorema 2.2.2 (Hiperplano de suporte) *Conforme [25], seja* $S \subset R^p$ *um conjunto convexo e* y *não pertence ao interior de* S. *Então existe um vetor* $u \neq 0$ *tal que :* $u^T x \geq u^T y$, *onde* x *é um ponto interior de* S.

Teorema 2.2.3 (Hiperplano de separação) *Conforme [25], sejam* S *e* T *convexos e* $S,\ T \subset \mathbb{R}^p$, $S \cap T = \varnothing$ *e* $S,\ T \neq \varnothing$, *então existe um hiperplano que separa* T *e* S *desde que exista* u *tal que* $\inf_{x \in S} u^T x \geq \sup_{x \in T} u^T x$.

Teorema 2.2.4 (Separação estrita) *Em [25], se* S *e* T *são convexos e* $S,\ T \subset \mathbb{R}^p$, $S \cap T = \varnothing$ *e* $S,\ T \neq \varnothing$; *tais que* S *é fechado e* T *é compacto, então existe um hiperplano que os separa estritamente, ou seja, um vetor* $u \neq 0$ *e um escalar tal que:*

$$u^T x_S < \alpha < u^T x_T \qquad \forall x_S \in S \quad e \quad x_T \in T$$

Definição 2.2.3 (Conjunto limitado) *Em [24], um conjunto S é limitado se existir um número real M e um ponto $q \in X$ tal que $d(p,q) < M$ para todo $p \in S$.*

Definição 2.2.4 (Ponto limite) *Em [24], um ponto "p" é um ponto limite de S se toda vizinhança de "p" contém um ponto $q \neq p$ tal que $q \in S$, [24].*

Definição 2.2.5 (Conjunto fechado) *Em [24], S é fechado se todo ponto limite de S é um ponto interior de S.*

Definição 2.2.6 (Conjunto compacto) *Em [24], S é um conjunto compacto em X, em espaços euclidianos, se e somente se for fechado e limitado.*

2.3 Funções Convexas

Convexidade é um conceito forte, entretanto, funções convexas não são necessariamente diferenciáveis; porém é possível obter um hiperplano de suporte que pode ser construído no ponto x para um suposto conjunto convexo acima do gráfico de f. Então as normais para o conjunto de possíveis hiperplanos para X contêm as propriedades do vetor gradiente de uma função convexa ajustada, conforme ilustrado na Figura 2.1(b). Devidamente normalizados estes conjuntos são chamados de subdiferenciais, [26].

Definição 2.3.1 (Funções convexas) *Em [24], seja S um conjunto convexo em $\mathbb{R}^p$. Então $f : \mathbb{R}^p \to \mathbb{R}$ é convexo sobre S se $f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y)$ onde para todo $x, y \in S$ e $0 \leq \theta \leq 1$, logo f é uma função convexa conforme ilustrado na Figura 2.1a.*

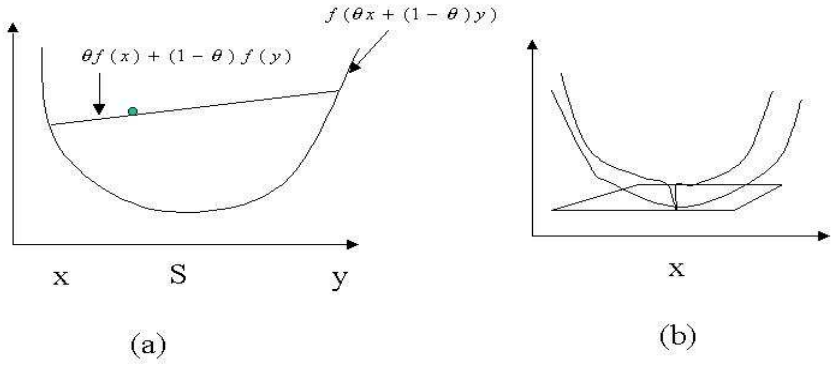

Figura 2.1: - (a) - Função convexa (b) - Cobertura convexa

É conveniente afirmar que f é limitada em S exceto quando $f \to -\infty$ se $\|x\| \to \infty$.

A função f é dita **côncava** se $-f$ é convexa. A função f é dita **estritamente convexa** se a desigualdade é estrita para todo $x, y \in S$ com $x \neq y$, e $\forall\ \theta \in (0, 1)$.

Como conseqüência obtém-se as seguintes afirmações:

- Para qualquer coleção $\{S_i \mid i \in I\}$ de conjuntos convexos, o conjunto interseção $\cap_{i \in I} S_i$ é convexo.

- O conjunto formado por todas as somas dos vetores $\{x_S + x_F | x_S \in S,\ x_F \in F\}$, de dois conjuntos convexos S e F, é convexo.

- A transformação linear da imagem de um conjunto convexo é outro conjunto convexo.

- Se S é um conjunto convexo e $f : S \to \mathbb{R}$ é uma função convexa. Então os conjuntos de níveis $\{x \in S \mid f(x) < \alpha\}$ e $\{x \in S | f(x) \leq \alpha\}$ são convexos para todo o escalar α.

Seja $S \subset \mathbb{R}^p$ um conjunto convexo e $f : S \to \mathbb{R}$ uma função diferenciável sobre S. Em resumo as seguintes proposições fornecem o meio de verificar a convexidade:

- A função f é convexa se e somente se: $f(y) \geq f(x) + \nabla f(x)^T(y - x)$ para $\forall x,\ z \in S$.

- Se a desigualdade for estrita sempre que $x \neq y$, então f é estritamente convexa.

- Se $\bigtriangledown^2 f(x)$ é positiva semidefinida para todo $x \in S$, então f é convexa.

- Se $\bigtriangledown^2 f(x)$ é positiva definida para todo $x \in S$, então f é convexa.

- Se $S \subset \mathbb{R}^p$ e f é convexa, então $\bigtriangledown^2 f(x)$ é positiva semidefinida $\forall x \in S$.

- A função $f(x) = x^T Q x$ é convexa se e apenas se Q é positiva definida.

- A função $f(x) = x^T Q x$ é estritamente convexa se e apenas se Q é positiva definida.

2.4 Condições de Otimalidade

Dado um problema de otimização deseja-se obter a caracterização da solução global, isto é, as condições necessárias e suficientes para um ponto viável ser um mínimo (máximo) global de uma função objetivo. Estas condições servem de base dos algoritmos de otimização, como mostrado em [27], e permitem além do reconhecimento dos

pontos como solução, determinar o comportamento numérico de vários algoritmos na proximidade da solução e ainda ajudam na análise de sensibilidade.

As condições de otimalidade aqui apresentadas são baseadas inicialmente nos critérios analíticos e em seguida nas condições necessárias para a convergência dos métodos estocásticos.

Devido à dificuldade do estabelecimento de critérios gerais que possam atender à todas as classes de problemas não lineares de otimização, divide-se estes problemas em duas grandes classes:

- Desvinculado (irrestrito)
- Vinculado (com restrição)

Dado um problema de otimização, interessa obter a solução global, isto é, necessita-se de uma condição suficiente para um ponto do conjunto de busca ser um mínimo (ou máximo), aplicando-se a expressão "minimização global" às questões relativas ao seguinte problema:

1. Encontrar o menor valor de f sobre todo S.
2. Encontrar o mínimo global de f sobre S, isto é $x^* \in S$ na qual $f(x^*) \leq f(x)$ para $\forall x \in S$.

O problema de minimização local tem o mesmo conjunto de objetivos como ponto de partida, porém é satisfeita com a obtenção de elementos x^* de S quando $f(x^*) \leq f(x)$ para todo $x \in S \cap N$, onde N é alguma vizinhaça de x^*. Então, na minimização global, se decide que a vizinhança de N é todo o espaço de busca.

Desta forma, pode-se observar as diferenças notáveis entre a otimização global e local. Usando-se as idéias chaves que governam os aspectos teóricos (primeira e

segunda ordem de aproximação das funções objetivo e restrições, linearização, penalização, teoria da dualidade, etc.) encontram-se numerosos algoritmos numéricos bem documentados na literatura. Por outro lado, nota-se em otimização global as seguintes características:

- Uma ampla variedade de idéias de **diferentes origens e natureza**: a dificuldade do na solução do problema conduz à observação dos processos físicos e da natureza que de algum possa servir de modelo para aquele observado;

- Dificuldades de **acesso a códigos numéricos**: os códigos que apresentam performance eficiente sobre problemas reais continuam confidenciais ou condizentes com as necessidades práticas no contexto para o qual teria sido designado. No presente existe um grande número de artigos voltados para otimização global, especialmente voltados para problemas de engenharia: centenas de artigos de otimização global voltados para um tipo particular de uma classe de problema.

O ponto onde começa a dificuldade está em distinguir, de maneira clara, o que é possível fazer ou não. Felizmente existem alguns artigos recentes e livros sintetizando o estado da arte com a classificação de problemas e meios de como resolvê-los, embora ainda incipientes.

O cálculo diferencial fornece importantes regras para a otimização local, mas, muito pouco para a otimização global. A razão é simples: a derivada de primeira ordem de uma função para um ponto inicial qualquer fornece uma direção de busca local. Para globalizar uma condição local baseada em cálculo diferencial, é necessário algum conhecimento sobre a estrutura da função; o simples e melhor conhecimento é a **convexidade**. A partir destes dois aspectos teóricos procura-se identificar de

que forma e situações a estrutura do problema fornece a caracterização das soluções globais.

Baseando-se no cálculo diferencial apresentam-se a seguir as premissas que fornecem as condições de otimalidade.

Teorema 2.4.1 *Supondo-se que* x^* *e* x *são pontos no* $\mathbb{R}^n$ *e que* $f : \mathbb{R}^n \to \mathbb{R}$ *é uma função duas vezes diferenciável sobre um conjunto aberto contendo o segmento* $[x^*,\ x] = \{w \in \mathbb{R}^n \mid w = x^* + t(x - x_o),\ 0 \leq t \leq 1\}$ *que une os pontos* x^* *e* x. *Então existe um ponto* $z \in [x^*,\ x]$ *tal que:*

$f(x) = f(x^*) + \nabla f(x^*)^T(x - x^*) + \frac{1}{2}(x - x^*)^T\nabla^2 f(z)(x - x^*)$ onde $\nabla f : \mathbb{R}^n \to \mathbb{R}^m$ é o gradiente e $\nabla^2 f : \mathbb{R}^n \to \mathbb{R}^{m\times n}$ é a Hessiana de f.

Logo, considerando as pequenas variações Δx de um dado vetor x^* obtém-se:

$f(x^* + \Delta x) \approx f(x^*) + \nabla f(x^*)^T\Delta x$ fornece a variação de $1^{\underline{a}}$ ordem;

e $f(x^* + \Delta x) \approx f(x^*) + \nabla f(x^*)^T\Delta x + \frac{1}{2}\Delta x^T\nabla^2 f(x^*)\Delta x$ fornece a variação da função objetivo até $2^{\underline{a}}$ ordem.

Se x^* é um mínimo local irrestrito de f, então a variação da função objetivo de primeira ordem, devido a uma pequena variação Δx, é não-negativa, ou seja:

$$\nabla f(x^*)^T\Delta x = \sum_{i=1}^{n} \frac{\partial f(x^*)}{\partial x_i}\Delta x_i \geq 0$$

Definindo-se Δx como múltiplos positivos e negativos dos vetores de coordenadas unitárias obtém-se, respectivamente:

$$\frac{\partial f(x^*)}{\partial x_i} \geq 0$$

$$\text{e } \frac{\partial f(x^*)}{\partial x_i} \leq 0$$

para $i = 1, 2.....n$. Então, dado que Δx é arbritário, tem-se: $\nabla f(x^*) = 0$.

Teorema 2.4.2 *Seja x^* um ponto interior de S no qual f tem um mínimo local. Se f é diferenciável em x^* então:*

$$\nabla f(x^*) = 0$$

Define-se um vetor x^* que satisfaz à condição acima como um ponto estacionário de f.

Teorema 2.4.3 *Seja x^* um ponto interior de S no qual f é duas vezes continuamente diferenciável. Se:*

$$\nabla f(x^*) = 0$$

$$e \ \ \Delta x^T \nabla^2 f(x^*) \Delta x \geq 0 \ \forall \Delta x \neq 0$$

então a matriz $\nabla^2 f(x^*)$ é positiva semidefinida.

Até agora foi verificado o comportamento da função em x^*. Verificando-se que em alguma vizinhança do mínimo local pode-se obter algumas condições adicionais para o mínimo global.

Teorema 2.4.4 *Em [24], seja um ponto interior de S e assumindo f duas vezes continuamente diferenciável em S. Para que exista um mínimo local de f em x^* é necessário que:*

$$\nabla f(x^*) = 0 \ e \ \Delta x^T \nabla^2 f(x^*) \Delta x > 0$$

Então a variação de segunda ordem de f devido a uma pequena variação Δx é positiva. Assim, f tende a crescer estritamente com pequenas excursões a partir de x^*, seguindo que as condições $\nabla f(x^*) = 0$ e $\nabla^2 f(x^*) > 0$ são suficientes para otimalidade local de x^*. Mínimos locais que não satisfazem às condições de suficiência são ditos pontos singulares, caso contrário, não-singulares.

Mínimos locais singulares são de difícil tratamento devido à ausência de convexidade de f, assim as otimalidades dessas funções não podem ser asseguradas usando as condições de suficiência e ainda na vizinhança do ótimo, o comportamento do algoritmo de otimização comumente utilizado (baseado em derivadas) tende a ser lento ou errôneo.

O problema de otimização vinculado é encontrado com dois tipos de restrições: igualdade e desigualdade; o problema é:

$$\text{Min } f(x)$$

$$\text{Sujeito à } h_i(x) = 0 \quad i = 1, 2...., m$$

$$g_j(x) \geq 0, \quad j = m + 1, ..., p$$

onde $x \in \mathbb{R}^n$.

Considerando o caso em que $g_j(x) = 0, \ \forall j = m+1...p$. Onde só existem restrições de igualdade assumir as hipóteses a seguir:

1. $h_i \in S, \quad i = 1, ...m \quad$ onde $S \subset \mathbb{R}^p$

2. Os gradientes $\nabla h_i(x^*), \ i = 1, ..., m$ são linearmente lndependentes, conforme descrito em [28] e [29].

Teorema 2.4.5 *Uma condição necessária para que o ponto* $x^* \in \mathbb{R}^n$*, satisfazendo as hipóteses 1 e 2, seja um mínimo local de uma função* f *continuamente diferenciável em* x^* *é que exista multiplicadores de Lagrange* $\lambda_1, ..., \lambda_m$ *tal que:*

$$\nabla f(x^*) - \sum_{i=1}^{m} \lambda_i \nabla h_i(x^*) = 0$$

$$\lambda_i \geq 0 \quad e \quad \lambda_i h_i(x^*) = 0 \quad para \quad i = 1..., m$$

Essa é uma condição necessária de primeira ordem, também conhecida como condição de Karush-Kuhn-Tucker.

Teorema 2.4.6 *Considera-se que* $x^* \in S \subset \mathbb{R}^n$ *e que as hipóteses 1 e 2 valem para* x^**. E ainda que* $f,\ h_1, ...h_m \in S$*. Então se* x^* *é um mínimo local de* f*, existem reais* $\lambda_1, ...\lambda_m$ *tais que:*

$$\nabla L(x, \lambda) = \nabla f(x^*) - \sum_{i=1}^{m} \lambda_i \nabla h_i(x^*) = 0$$

$$\lambda_i \geq 0 \quad e \quad \lambda_i h_i(x^*) > 0$$

$$\Delta x^T H(x^*, \lambda) \Delta x > 0, \forall \Delta x \in M = \{y \in R^n : \nabla h(x^*) y = 0\}$$

onde H é a matriz Hessiana de $L(x, \lambda)$ com relação a x.

$H(x, \lambda) = \nabla^2 f(x) + \sum_{i=1}^{m} \lambda_i \nabla^2 h_i(x)$ e M é um subespaço definido como $M = \{y \in \mathbb{R}^n : \nabla h(x^*) y = 0\}$.

Com isto verifica-se que as estimativas de programação não-linear com restrição são tradicionalmente resolvidas por acréscimo à função objetivo ou à correspondente

função de Lagrange usando penalidades ou termos barreiras para contabilizar as restrições, conforme descrito em [30] e [31].

A função resultante de mérito é então otimizada usando qualquer procedimento de otimização irrestrita. Independentemente da técnica usada, a função de mérito sempre depende de um parâmetro x_i. Como $x_i \to 0$ minimizadores da função de mérito convergem para o conjunto de minimizadores da função original.

Devido à importância da função objetivo ser convexa, serão realizadas algumas considerações sobre convexificação de uma função não convexa; toda a função convexa que tenha um mínimo local, este mesmo mínimo também é global.

Definição 2.4.1 (Cobertura de Função Não Convexa) *Fazendo* $f : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ *ser uma função não convexa que satisfaz as seguintes propriedades:*

$$\text{Dom}\, f = \{x \in \mathbb{R}^n \mid f(x) < +\infty\} \quad \text{é não vazio} \tag{2.4.1}$$

$$\lim_{x^* \to x} \inf f(x^*) \geq f(x) \forall x \in \mathbb{R}^n \tag{2.4.2}$$

$$f \quad \text{é l - coercivo sobre } \mathbb{R}^n \; se \lim_{\|x\| \to +\infty} \frac{f(x)}{\|x\|} = +\infty \tag{2.4.3}$$

O conjunto fechado convexo relativo de f é chamado de uma cobertura convexa (ou envelope convexo) co de f, na qual pode ser definida de diversas formas: para todo $x \in \mathbb{R}^n$.

$$(\text{co}\, f)(x) = \text{Sup}\, \{g(x) | g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\} \; convexo, \; g \leq f\} \tag{2.4.4}$$

ou seja, $(\text{co}\, f)$ é o supremo de todas as funções afins minimizadas de f:

$$(\text{co}\, f)(x) = inf\{\sum_{i=1}^{n+1} \alpha_i f(x_i) \; tal \; que \; (\alpha_1, ..., \alpha_{n+1}) \in \Delta_{n+1}, \; x_i \in \text{dom}(f), \; \sum_{i=1}^{n+1} \alpha_i x_i = x\} \tag{2.4.5}$$

onde Δ_{n+1} é a unidade do simplex de $\mathbb{R}^{n+1}$, que é:

$$\Delta_{n+1} = \{(\alpha_1, ..., n+1) \in \mathbb{R}^{n+1} \mid \sum_{i=1}^{n+1} \alpha_i = 1, \alpha_i \geq 0 \text{ para } i = 1, ...n+1\} \quad (2.4.6)$$

A equivalência entre as três definições acima de $(\operatorname{co} f)$ é principalmente devido a equação 2.4.3 além disso $\operatorname{co} f$ é a menor função semicontínua sobre $\mathbb{R}^n$ e para algum $x \in \operatorname{Dom}(\operatorname{co} f) = \operatorname{co}(\operatorname{Dom} f)$ existirá $x_i \in \operatorname{Dom}(f)$ e $(\alpha_1, ..., \alpha_{n+1}) \in \Delta_{n+1}$ tal que:

$$x = \sum_{i=1}^{n+1} \alpha_i x_i, \quad e \quad (\operatorname{co} f)(x) = \sum_{i=1}^{n+1} \alpha_i f(x_i). \quad (2.4.7)$$

Fornecendo as seguintes propriedades:

Para um dado $x \in \operatorname{co}(\operatorname{dom} f)$, a subfamília $\{x_i\}_{i \in I}$ de vetores $\{x_1, ..., x_{n+1}\}$ descrita na equação 2.4.7 correspondente a valores positivos α_i $(i \in I$ onde $\alpha_i > 0)$ isto é "chamado de x ". Se $x_{i \in I}$ é chamado de x, então:

$$f(x) = (\operatorname{co} f)(x_i), \ \forall i \in I \quad (2.4.8)$$

$\operatorname{co} f$ é uma função afim sobre o poliédro compacto (convexo)

$$\operatorname{co}\{x_i | i \in I\} \quad (2.4.9)$$

Se, além disso, f é diferenciável sobre alguns x_i, então $(\operatorname{co} f)$ é diferenciável sobre x e

$$\langle \nabla(\operatorname{co} f)(x), \ x\rangle - (\operatorname{co} f)(x) = \langle \nabla f(x_i), \ x_i\rangle - f(x_i) \quad (2.4.10)$$

Satisfatoriamente suficiente, se $f : \mathbb{R} \to \mathbb{R}$ é diferenciável (e limitado inferiormente por alguma função afim), igualmente é $(\operatorname{co} f)$. O resultado é falso se f é uma função de muitas variáveis; entretanto uma l-coerciva função diferenciável $f : \mathbb{R}^n \to \mathbb{R}$ produz uma diferenciável $(\operatorname{co} f)$.

Agora como associar a minimização de f com da $\operatorname{co} f$? Para f satisfazer as equações (2.4.2), (2.4.3) e (2.4.4), a menor fronteira de f é finita e alcançável, e o conjunto $\operatorname{Argmin} f$ dos pontos mínimos é um conjunto compacto não vazio.

Teorema 2.4.7 *A partir das afirmações fornecidas pelas equações (2.4.2), (2.4.3) e (2.4.4) sobre f, segue-se que são verdadeiras as afirmações:*

$$\operatorname*{Min}_{x\in\mathbb{R}^n} f(x) \;=\; \operatorname*{Min}_{x\in\mathbb{R}^n} (cof)(x) \tag{2.4.11}$$

$$\operatorname{Argmin}(\operatorname{co} f) \;=\; \operatorname{co}(\operatorname{Argmin} f) \tag{2.4.12}$$

Conseqüentemente algum minimizador de $\operatorname{co} f$ é uma combinação de minimizadores globais convexos de f. Com um colorário imediato:

$$\overline{x} \text{ minimiza } f \text{ globalmente} \Leftrightarrow x^* \text{ minimiza } \operatorname{co} f \;\; e \;\; (\operatorname{co} f)(x^*) \;=\; f(x^*). \tag{2.4.13}$$

Sabe-se que para um dado x^* minimizar uma função f diferenciável, a condição necessária, mas não suficiente, é que $\nabla f(x^*) \;=\; 0$ (onde x^* é um ponto crítico ou estacionário). A questão natural é: qual a condição adicional para assegurar que um ponto estacionário é um global de f ? Certamente a condição procurada deve atender à condição global. Isto envolve o comportamento de f sobre o espaço viável. A convexificação de f torna -se uma ferramenta apropriada para se obter uma condição.

Teorema 2.4.8 *Sendo $f : \mathbb{R}^n \to \mathbb{R}$ diferenciável sobre $\mathbb{R}^n$, então x^* é um mínimo global de f sobre $\mathbb{R}^n$, se e somente se:*

$$\text{(i)} \quad \nabla f(x^*) \;=\; 0 \tag{2.4.14}$$

e

$$\text{(ii)} \quad (\text{co}\, f)(x^*) \;=\; f(x^*) \tag{2.4.15}$$

Deste modo, $\text{co}\, f$ é diferenciável em x^* e $\nabla\,(cof)(x^*) \;=\; 0$.

Portanto, a propriedade global (ii) é justamente o que é procurado na propriedade local (i) para um ponto estacionário ser um ponto de mínimo global. Uma conclusão, por exemplo, é que funções diferenciáveis, cujos pontos estacionário são mínimos globais, são funções f nas quais:

$$\{x^* \mid \nabla f(x^*) \;=\; 0\} \;\subset\; \{x^* \mid (cof)(x^*) \;=\; f(x^*)\} \tag{2.4.16}$$

Observa-se que a condição (ii) não é fácil de se obter. Isto é dificultado pelo cálculo de $\text{co}\, f(x^*)$ exato.

Uma das tentativas de se desvencilhar das derivadas sobre f é fazendo a substituição $0 \;\in\; \overset{\wedge}{\nabla} f(x^*)$ por $\nabla f(x^*) \;=\; 0$, onde $\overset{\wedge}{\nabla} f$ é o conjunto de valores da generalização da noção do gradiente. Exemplos simples de uma dimensão uma equivalência do Teorema (2.4.8) (em uma versão adaptada da condição (i)) é atendida se não se assumir que f é diferenciável para x^*.

Para uma classe de funções objetivo (não necessariamente diferenciáveis), sob as condições descritas abaixo, tornam-se diferenciáveis no mínimo global e os resultados do Teorema (2.4.8) serão válidos.

Considerar os seguintes tipos de funções:

$$f \;=\; \inf_{t \in T} f_t(x), \quad \textit{onde}\ f_t \;:\; \mathbb{R}^n \to \mathbb{R} \ \text{é diferenciável para todo } t \;\in\; T. \tag{2.4.17}$$

assume-se também que para cada x, o conjunto de índices t para o qual $f_t(x) \;=$

$f(x)$ é não vazio, isto é:

$$T(x) \;=\; \{t \;\in\; T \mid f_t(x) \;=\; f(x)\} \;\neq\; \emptyset \tag{2.4.18}$$

Este é o caso de f ser a função mínima de um número finito de funções, isto é, $T \;= \{1, ..., k\}$.

Teorema 2.4.9 *Sendo f uma função do tipo como definida Equação(2.4.17) e assumindo que Equação (2.4.18) é satisfeita. Então, as seguintes afirmações são equivalentes:*

(i) x^* é um mínimo global de f sobre $\mathbb{R}^n$

(ii)f é diferenciável para x^*, $\nabla f(x^*) \;=\; 0$ e $(\mathrm{co}\, f)(x^*) \;=\; f(x^*)$.

Para provar apoia-se na teoria da existência de subdiferenciais para funções não-convexas.

Dada uma função arbritária $f : \mathbb{R}^n \to \mathbb{R}$, a subdiferencial $\partial f(x^*)$ de f em x^* (no ambiente de análise convexa) é o conjunto de $s \;\in\; \mathbb{R}^n$ satisfazendo:

$$f(x) \;\geq\; f(x^*) \;+\; \langle s,\; x \;-\; x^* \rangle \quad \text{para todo} \quad x \;\in\; \mathbb{R}^n \tag{2.4.19}$$

Certamente a condição $0 \;\in\; \partial f(x^*)$ é necessária e suficiente para x^* ser um mínimo global de f sobre $\mathbb{R}^n$. Esta noção de subdiferencial ser uma condição global, não desperta muito interesse quando f não é convexa. Entretanto, as regras de cálculo são úteis.

Toma-se $f : \mathbb{R}^n \to \mathbb{R}$

1. Sendo $\partial f(x^*)$ não vazia, são equivalentes as seguintes propriedades:

$$(\mathrm{co}\, f)(x^*) \;=\; f(x^*) \;\text{ e, do mesmo modo, }\; \partial(\mathrm{co}\, f)(x^*) \;=\; \partial\, f(x^*) \tag{2.4.20}$$

2. Se f é diferenciável em x^*, então $\partial f(x^*)$ é vazio ou $\partial f(x^*) = \{\nabla f(x^*)\}$ neste caso:

$$\partial\,(\mathrm{co}\, f)(x^*) = \{\nabla f(x^*)\} \tag{2.4.21}$$

3. Se f é convexa, $\partial f(x^*)$ é um elemento de $\{s^*\}$ se e somente se f for diferenciável no ponto x^*, neste caso $s^* = \nabla f(x^*)$.

4. Se $f = \inf_{t\in T} f$, e se $T(x^*)$ é não vazio, então

$$\partial\, f(x^*) \subset \bigcap_{t\in T(x^*)} \partial\, f_t(x^*). \tag{2.4.22}$$

Retornando ao Teorema (2.4.8), tem-se:

$[(ii) \Rightarrow (i)]$ de acordo com as propriedades (1) e (2) acima $\partial(\mathrm{co}\, f)(x^*) = \partial f(x^*) = \{0\}$. Daí, $f(x) \geq (\mathrm{co}\, f)(x) \geq (\mathrm{co}\, f)(x^*) \geq f(x^*)$, $\forall x \in \mathbb{R}^n$.

$[(i) \Rightarrow (ii)\,]$ reescrevendo a propriedade (i) como $0 \in \partial f_t(x^*)$, sabe-se pela propriedade (1) acima que $(\mathrm{co}\, f)(x^*) = f(x^*)$. Resta provar que f é diferenciável em x^*. Como conseqüência da propriedade (4), $0 \in \partial f_t(x^*)\ \forall t \in T(x^*)$, na qual com a diferenciabilidade de f_t a propriedade (2) acima, implica:
$\partial f_t(x^*) = \{\nabla f_t\,(x^*)\} = \{0\}$ para todo $t \in T(x^*)$ e portanto, $\partial f_t(x^*) = \{0\}$. Porém, esta condição não é suficiente para assegurar que f é diferenciável em x^*. Por (1) $\partial(cof)(x^*)$ é também $\{0\}$, conseqüentemente (pela propriedade (3) acima) $\mathrm{co}\, f$ é diferenciável em x^* com $\nabla(\mathrm{co}\, f)(x^*) = 0$. Agora, f está colocado entre cof e f_t, $t \in T(x^*)$, ambos diferenciáveis em x^* e coincidindo em x^*, é propriamente dito diferenciável e $\nabla f(x^*) = 0$, pela definição de diferenciabilidade.

A convexificação de uma função torna-se um natural problema variacional: minimizando uma função objetivo da forma de $\int_a^b \ell(t,\ x(t))dt$ é associado à minimização de problema (menos rigoroso) da forma de $\int_a^b \mathrm{co}\,\ell(t,\ x(t))dt$ onde $co\ell$ é uma cobertura da função parcial $\ell(t,\ x)$.

Certamente, a generalização oferecida pelo Teorema (2.4.8) é devido à feliz combinação do processo de minimização com as funções do tipo Inf. Para as funções do tipo Sup um mínimo mais apropriado é uma dobra de f (um ponto onde f não é diferenciável).

2.5 Otimização de Diferenças Convexas

Convexidade (ou variantes como quasi-convexidade e pseudo-convexidade) de uma função objetivo é uma importante característica para um mínimo local ser global. A informação local como $\nabla f(\overline{x}) = 0$ torna-se global porque a convexidade possibilita a propagação da propriedade na vizinhança de $\overline{x}$ para todo o espaço viável. A minimização não-convexa está presente em muitas estruturas de problemas nas quais são de difíceis soluções. São consideradas aqui duas classes de problemas:

- ($\wp_1$) Minimiza a diferença de funções convexas (chamada de função d.c) sobre um conjunto convexo fechado (chamados de problemas de minimização d.c);

- ($\wp_2$) Maximiza a função convexa sobre um conjunto convexo (maximização convexa).

Atualmente estes dois problemas são equivalentes no sentido de que um pode ser transformado na mesma estrutura do outro. Entretanto, é interessante tratar os dois

problemas distintamente para se obter as condições de otimalidade. A característica comum da formulação $(\wp_1)$ e $(\wp_2)$ é a convexidade.

2.5.1 Minimização Global - Método das Diferenças Convexas DC

Sendo a função objetivo f da seguinte forma: $f = g - h$, onde $g : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ é a menor função subcontínua convexa. Para deduzir a condição necessária e suficiente de minimilidade global é necessário uma generalização do conceito de subdiferencial em otimização global, como já definido:

$$f(x) \geq f(\overline{x}) + \langle s, x - \overline{x} \rangle \text{ para todo } x \in \mathbb{R}^n. \tag{2.5.1}$$

Dada uma função arbitrária $\varphi : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$, $\overline{x}$ é um ponto para a qual φ é finita, e $\varepsilon \geq 0$, o ε - subdiferencial $\partial_t \varphi(\overline{x})$ de φ em $\overline{x}$ é um conjunto de $s \in \mathbb{R}^n$ satisfazendo:

$$f(x) \geq f(\overline{x}) + \langle s, x - \overline{x} \rangle \text{ para todo } x \in \mathbb{R}^n. \tag{2.5.2}$$

Teorema 2.5.1 (Existência do Mínimo Global) *Conforme [25], x é um mínimo global de $f = g - h$ se e somente se:*

$$\partial_\varepsilon h(\overline{x}) \subset \partial_\varepsilon g(\overline{x}) \quad \forall \varepsilon > 0 \tag{2.5.3}$$

Prova do Teorema do Mínimo Global.

Supondo $\overline{x}$ não ser um mínimo global de $f = g - h$, isto é, existe $\widetilde{x}$ tal que $g(\widetilde{x}) - g(\overline{x}) < h(\widetilde{x}) - h(\overline{x})$. Desta maneira pretende-se que o ponto $(\widetilde{x}, g(\widetilde{x}) - g(\overline{x}))$

não pertença à função $\widetilde{h} : x \mapsto \widetilde{h}(x) - h(x^*)$. Portanto, existe um hiperplano $\mathbb{R}^n$x$\mathbb{R}$ separando $(\widetilde{x},\ g(\widetilde{x}) - g(\overline{x}))$ de uma função $\widetilde{h}$ para algum (s, ε).

$$g(\widetilde{x}) - g(\overline{x}) < \langle s, \widetilde{x} - \overline{x} \rangle - \varepsilon. \tag{2.5.4}$$

$$h(x) - h(\overline{x}) \geq \langle s, x - \overline{x} \rangle - \varepsilon,\ \forall\ x \in\ \mathbb{R}^n. \tag{2.5.5}$$

Sendo $\varepsilon \geq 0$ (fazendo $x = \widetilde{x}$ na Equação(2.5.5) e $s \in \partial_\varepsilon h(\overline{x})$ porém na Equação(2.5.4) $s\ \notin\ \partial_\varepsilon g(\overline{x})$.

Como se vê na prova, a convexidade de g não é hipótese necessária, porém é uma condição formal desde que em um mínimo global $\overline{x}$ necessariamente tenha $g^*(\overline{x}) = g(\overline{x})$, conseqüentemente $\partial_\varepsilon g^*(x) = \partial_\varepsilon g(x)$ para todo $\varepsilon \geq 0$.

2.5.2 Maximização Convexa

Considere o problema de minimização global da função convexa do tipo $h : \mathbb{R}^n \rightarrow \mathbb{R}$ sobre um conjunto fechado convexo C. É claro que $\overline{x} \in C$ maximiza globalmente $h(x)$ sobre C se e somente se maximizar $f(x) = I_C(x) - h(x)$ sobre $\mathbb{R}^n$, onde $I_C(x) = 0$, se $x \in C$, e $+\infty$ caso contrário. Portanto, recai em um problema de minimização de diferenças convexas onde a função objetivo $f = g - h$ é de um tipo especial, nomeada $g = I_C$. Para um dado g tem-se:

$\partial_\varepsilon g(\overline{x}) = \partial_\varepsilon I_C(\overline{x}) = \{x \in \mathbb{R}^n \mid ,\ \overline{x}\} \leq \varepsilon,$ para todo $x \in C\}$

este conjunto (fechado e convexo) é denotado por $N_\varepsilon(C,\ \overline{x})$ e é chamado conjunto de ε - direção normal em C para $\overline{x}$.

Uma reformulação do Teorema (2.5.1) de maneira a atender a este contexto fica agora da seguinte forma.

Teorema 2.5.2 *Em [25], $x \in C$ é um máximo global de h sobre C se e somente*

se:

$$\partial_\varepsilon h(\overline{x}) \subset N_\varepsilon(C,\ \overline{x}), \quad \textit{para todo}\ \ \varepsilon > 0. \tag{2.5.6}$$

Se h (convexa ou não) é diferenciável em $\overline{x}$, uma condição necessária para $\overline{x} \in C$ ser um máximo local de h sobre C é $\nabla h(\overline{x}) \in N(C,\ \overline{x})$. Se h (não necessariamente diferenciável em $\overline{x}$) é convexa, a condição necessária torna-se $\partial h(\overline{x}) \subset N(C,\ \overline{x})$. Isto é precisamente o caso limite para $\varepsilon \geq 0$ na Eq(2.5.6). A globalização para todos os $\varepsilon \geq 0$ através da Eq(2.5.6) é uma condição necessária e suficiente para Maximalidade Global.

Naturais questões surgem com o Teorema(2.5.2): para que valores particulares de C e h a equação(2.5.6) faz sentido? Como pode uma estrutura especial de C e h simplificar a equação(2.5.6)? Como se pode usar a estrutura da equação(2.5.6) para projetar um algoritmo?

A estrutura que pode oferecer uma grande simplificação é quando h é quadrática (isto é, $h(x) = \frac{1}{2}\langle Ax,\ x\rangle + \langle h,\ x\rangle$, onde $A \in S_n(R)$ é positiva semidefinida) e C é poliedral.

Desta forma fica explicado de maneira rápida como a condição local pura $\partial h(\overline{x}) \subset N(C,\ \overline{x})$ torna global $\partial_\varepsilon h(\overline{x}) \subset N_\varepsilon(C,\ \overline{x})$ com a ajuda do parâmetro $\varepsilon > 0$. Isto torna possível outras formas de se obter as condições de otimização local pela consideração dos pontos x no nível - superfície de h no nível de $h(\overline{x})$. Esta abordagem é feita por A. Strekalovsk em uma série de artigos [32], onde são propostos e desenvolvidas as condições necessárias e suficientes (para maximalidade global) de uma outra forma. A idéia de forma simplificada está centrada em considerar o problema de minimização de uma função convexa $h : \mathbb{R}^n \to \mathbb{R}$ sobre um conjunto não vazio convexo fechado C. Retomando-se as seguintes definições:

- O nível- superfície de h no nível r é $\{x \in R^n \mid h(x) = r\}$.

- Se $x \notin C$, ainda define o cone normal para C no x (denotado por $N(C,\ x)$) como o conjunto de $d \in \mathbb{R}^n$ satisfazendo $\langle d,\ c - x\rangle \leq 0$ para todo $c \in C$.

Teorema 2.5.3 *Em [25], considerando-se os pressupostos acima, toma-se um ponto* $\overline{x} \in C$ *de maneira que* $-\infty \leq \inf_{\mathbb{R}^n} h < h(\overline{x})$. *Então* $\overline{x}$ *é um máximo global de* h *sobre* C *se somente se:*

$$\partial h(x) \subset N(C,\ x) \text{ para todo } x \text{ satisfazendo } h(x) = h(\overline{x}) \quad (2.5.7)$$

Prova (de Strekalovski)

Passo 1 - Inicia-se provando o seguinte Lema: se C e D são (não-vazios) conjuntos fechados convexos, então $C \subset D$ se e somente se $N(D,\ x) \subset N(C,\ x)$ para todo $x \in$ fronteira de D.

A condição é necessária fazendo $x \in$ fronteira de D e $d \in N(D,\ x)$ de maneira que $\langle d,\ u - x\rangle \leq 0$ para todo $u \in D$ e $D \supset C$, tem-se $\langle d,\ v - x\rangle \leq 0$ para todo $v \in C$, isto é, $d \in N(C,\ x)$.

Considerando que a condição anunciada é supostamente verdadeira e algum valor u em C porém não em D, fazendo α ser a distância de u até D, conseqüentemente existe x sobre a vizinhança de D de maneira que $0 < \alpha = \|u - x\|$.

Fazendo ϑ ser uma bola aberta (de raio α) centrada em u. De maneira que $\vartheta \cap D = \emptyset$, separando ϑ de D existe $s \neq 0,\ \gamma \in R$ de modo que:

$\langle s,\ v\rangle \leq \gamma < \langle s,\ a\rangle$ para todo $v \in D$ e $a \in \vartheta$.

Tem-se $\langle s,\ x\rangle \leq \gamma$ porque $x \in D : \langle s,\ x\rangle \geq \gamma$ porque x é sustentável sobre a fronteira de ϑ . Por isso $\langle s,\ x\rangle = \gamma$. Então

$\langle s,\ v - x\rangle = \langle x,\ v\rangle - \langle s,\ x\rangle \leq \gamma - \gamma = 0$ para todo $v \in D$, isto é , $s \in N(D,)$.

Agora, já que $u \in \vartheta \cap C,\ \langle s,\ u\rangle > \gamma$, que é $\langle s,\ u - x\rangle > 0$. Conseqüentemente, $u \notin N(C,)$ que é uma contradição.

Passo 2

Esclarecendo, $\overline{x} \in C$ é um máximo global de h sobre C se e somente se

$$C \subset \{x \in R^n \mid h(x) \leq h(\overline{x})\} = D \tag{2.5.8}$$

Porque o $\underset{\mathbb{R}^n}{inf}\ h < h(\overline{x}),$ por hipótese, existe algum $\widetilde{x}$ de maneira que $h(\widetilde{x}) < h(\overline{x})$. Conseqüentemente na fronteira de D está o nível - superfície $\{x \in \mathbb{R}^n \mid h(x) = h(\overline{x})\}$, e $N(D,\ x) = \mathbb{R}^+\partial h(x)$ para todo $x \in$ fronteira de D. Deste modo, segue o resultado do passo 1 e a inclusão da equação(2.5.7) é equivalente a:

$\mathbb{R}^+\partial h(x) \subset N(C, x)$ para todo x tal que $h(x) = h(\overline{x})$.

2.6 Condição de Otimalidade Global Baseada em Integração

Nesta seção são apresentados alguns resultados de otimização global, baseados em integração. Estes procedimentos tratam funções objetivo contínuas e conjuntos viáveis compactos. Na realidade prática, muitas vezes estas condições não são atendidas, entretanto, as funções objetivo (sob determinadas condições) podem ser convexificadas sem alterar o sentido do problema original e ainda alguns resultados teóricos de interesse são obtidos.

Assumindo que o conjunto de restrições S é um fecho de uma fronteira não vazia, fronteira de um conjunto aberto(um exemplo é um conjunto compacto com interior não vazio) e $f : S \to \mathbb{R}$ é uma função contínua. Utilizando os conceitos de grandes

desvios de probabilidade e estatística propostos por Laplace, em [10], mostra-se que:

$$\mathrm{Max}_{x\in S}\, f(x) = \lim_{k\to+\infty}\left[\frac{1}{k}\log\int_S e^{kf(x)}dx\right] \tag{2.6.1}$$

As questões agora são: Como fazer uma aproximação eficiente da integral acima? O procedimento fica estável quando $k \to +\infty$? Se uma função contínua g estritamente positiva sobre S, da transformação $f = \log g$ for aplicada à equação(2.5.3) obtém-se a seguinte expressão que representa uma seqüência de valores:

$$r_k = \int_S e^{\frac{g(x)}{g(\overline{x})}k}dx, \quad \overline{x} \in S \tag{2.6.2}$$

onde $\overline{x}$ é um máximo global de g sobre S se e somente se $\{r_k(\overline{x})\}$ for limitada.

Um resultado semelhante associado à seqüência de aproximação de um maximizador global de g sobre S é proposto a seguir.

Teorema 2.6.1 *Conforme [25], tomando S como um conjunto fechado com uma fronteira não vazia aberta e $f : S \to \mathbb{R}$ contínua, supondo f ser globalmente maximizada sobre S por um único ponto $\overline{x}$, então a seqüência:*

$$\overline{x} = \frac{\int_S xe^{kf(x)}dx}{\int_S e^{kf(x)}dx} \tag{2.6.3}$$

converge para $\overline{x}$ onde $k \to +\infty$.

Supondo agora $f : \mathbb{R}^n \to \mathbb{R}$ ser continua e 0-coerciva (isto é $\lim_{\|x\|\to+\infty} f(x) = +\infty$), de maneira que os conjuntos-subníveis são compactos. Para $r > \mathrm{Max}_{\mathbb{R}^n} f$, o valor médio $\mu_r(r)$ é definido via a limitação do processo: $\lim_{r_k\to\mu_k(r_k)}$.

Então $\overline{x}$ é um mínimo global de f sobre $\mathbb{R}^n$ se e somente se :

$$\mu_r[f(\overline{x})] = f(\overline{x}) \tag{2.6.4}$$

Esta estimativa é desenvolvida em detalhes em [33]. Entretanto, as dificuldades inerentes para integração eficiente sobre conjuntos-subníveis (pode ser razoavelmente complicada) em razão das generalizações sobre as funções objetivo consideradas funções contínuas. Por estas razões há dúvida sobre a operacionalidade do algoritmo.

Nesta avaliação dos algoritmos atuais, da classe dos determinísticos, foram apresentados os mais promissores e efetivos para otimização global. Entretanto, deve-se considerar também alguns outros resultados específicos, por exemplo em otimização polinomial: quando a função objetivo é polinomial, a situação onde há um mínimo local (ou algum ponto estacionário) é necessariamente global? A resposta depende de um caminho específico tomado sobre o número de variáveis e do grau p do polinômio. Alguns resultados com este objetivo podem ser obtidos em [34].

Como visto, não foi encontrada uma condição necessária e suficiente de otimalidade global para todas as classes de problemas de otimização. Entretanto, alguns problemas especialmente estruturados podem ser instrumentos utilizados para obter algum resultado. A mensagem é contudo clara: para tratar otimalidade global, é necessário conhecer mais matemática, que possivelmente mude as estimativas , e desenvolva novas ferramentas.

2.7 Critérios de Otimalidade para Algoritmos

A convergência de um algoritmo pode ser garantida por meio da prova da existência de uma seqüência de pontos ao longo de uma direção de busca $p_k^i \ i = 1, ..., r$, satisfazendo à seguinte condição:

Axioma 2.7.1 (C1) *Como descrito em [35], dada uma seqüência limitada de pontos* $\{x_k\}$ *e de direções* $\{p_k^i\}$*, com* $i = 1, 2...., r$*, tal que :*

$$\lim_{k\to\infty} \|\nabla f(x_k)\| = 0$$
$$\lim_{k\to\infty} \sum_{i=1}^{r} \{0, \nabla f(x_k)^T p_k^i\} = 0$$

Nos algoritmos de busca direta o primeiro passo na definição da direção da busca é obter um conjunto de direções apropriadas $p_k^i, i = 1, ..., r$, onde cada ponto x_k é produzido pelo algoritmo. Este conjunto de direções deve ter um comportamento local, apropriado, sobre a função objetivo que gera informação suficiente para sobrepor a falta do gradiente.

A condição C1 caracteriza o conjunto de direções que satisfazem esta propriedade. Esta condição requer que a distância entre os pontos gerados pelo algoritmo e o conjunto de pontos estacionários da função objetivo tenda para zero se, e somente se, as derivadas direcionais da função objetivo ao longo das direções p_k^i, $i = 1, ..., r$, tenda a assumir valores não negativos. Considerando a tomada de uma direção qualquer, na qual o ponto corrente não é um ponto estacionário, é possível obter uma direção no qual a função objetivo decresça suficiente, usando o conjunto de direções que satisfazem a condição C1.

Em particular, pelo uso da condição C1, se pode caracterizar um ponto estacionário de f com o fato de que a função objetivo não decresce localmente ao longo de uma direção $\{p_k^i\}$, $i = 1, 2, ..., r$, em pontos sufucientemente próximos de um ponto corrente x_k. Isto leva à possibilidade de uma definição de uma condição geral para a convergência do método de busca direta por meio da existência de seqüências na qual a função objetivo decresça. Esta condição é inalterável, muito simples e intuitiva,

permitindo identificar muitos requisitos sobre aceitáveis amostras da função objetivo ao longo da direção da busca $\{p_k^i\}$, $i = 1, 2, ..., r$ que garante a convergência global do método.

Supondo a seguinte hipótese :

O conjunto de nível $L_0 = \{x \in \mathbb{R}^n : f(x) \leq f(x_0)\}$ é compacto. O seguinte Teorema descreve um conjunto de condições de convergência global:

Teorema 2.7.1 (Seqüências Convergentes) *Em [35], seja* $\{x_k\}$ *uma seqüência de pontos;* $\{p_k^i\}$ *,* $i = 1, ..., r$ *uma seqüência de direções, e supondo que suportam as seguintes condições:*

1. $f(x_{k+1}) \leq f(x_k)$
2. $\{p_k^i\}$, $i = 1, ...r$, *satisfazendo a condição C1*
3. *Exista pelo menos uma seqüência de pontos* $\{y_k^i\}$ *e de escalares positivos* $\{a_k^i\}$, $i = 1, ..., r$ *tal que:*

$$f(y_k^i + a_k^i p_k^i) \geq f(y_k^i) - o(a_k^i) \tag{2.7.1}$$

$$\lim_{k \to \infty} a_k^i = 0 \tag{2.7.2}$$

$$\lim_{k \to \infty} \|x_k - y_k^i\| = 0 \tag{2.7.3}$$

$$Então \quad \lim_{k \to \infty} \|(x_k)\| = 0 \tag{2.7.4}$$

Prova. De (1) e considerando que $\{f(x_k)\}$ é uma seqüência não crescente, tal que $\{x_k\}$ pertence a um conjunto compacto L_o e admitindo ao menos um ponto limite, faz-se x ser algum ponto limite $\{x_k\}$. Então, existe um subconjunto $K_1 \subseteq \{0, 1...\}$ tal que:

$$\lim_{k\to\infty, k\in K_1} x_k = \overline{x}$$

$$\lim_{k\to\infty, k\in K_1} p_k^i = \overline{x}^i \quad i = 1, ..., r$$

Usando a Equação(2.7.3) segue que:

$$\lim_{k\to\infty, k\in K_1} y_k^i = \overline{x}$$

Agora, tomando-se a Equação (2.7.1) para todo $k \geq 0$, tem-se:

$$f(y_k^i + a_k^i p_k^i) - f(y_k^i) \geq -o_k^i \quad i = 1, ..., r \tag{2.7.5}$$

Pelo Teorema do Valor Médio pode-se escrever:

$$f(y_k^i + a_k^i p_k^i) - f(y_k^i) = a_k^i \nabla f(u_k^i)^T p_k^i, \quad i = 1, ..., r \tag{2.7.6}$$

Agora é possível observar a partir da Equação (2.7.2), tomando a contagem na fronteira de $\{p_k^i\}$, que $u_k^i \to \overline{x}$ quando $k \to \infty$, $k \in K_1$. Conseqüentemente, para a continuidade de ∇f a partir da Equação (2.7.7) e tomando-se a equação (2.7.2), tem-se:

$$\lim_{k\to\infty, k\in K_1} \nabla f(u_k^i)^T p_k^i = \nabla f \overline{x}^T \overline{p_k} \geq 0, \quad i = 1, ...r$$

Então tomando-se (2) e a Condição (C1), tem-se: $\nabla f(x) = 0$

Como $\overline{x}$ é um ponto limite de $\{x_k\}$, conclui-se que:

$$\lim_{k\to\infty} \|\nabla f(x_k)\| = 0$$

Em linhas gerais de acordo com (3), para cada direção de busca p_k^i, a existência de pontos apropriados y_k^i e $y_k^i + a_k^i p_k^i$ associada a um ponto corrente x_k é assumido (veja a Equação (2.7.2 e 3)) sempre que ocorre uma falha na condição (suficiente) de decrescimento estrito de f(veja Equação (2.7.1)).

Então, considerando a Equação(2.7.2), tem-se que no ponto y_k^i a derivada direcional de f ao longo de p_k^i pode ser aproximada por uma quantidade na qual tende a ser não negativa. Entretanto, adequando a propriedade da direção da busca pela condição C1, a convergência global da seqüência $\{x_k\}$ pode ser assegurada pelo requisito dos pontos da nuvem falharem o mais próximo de x_k.

O uso da direção satisfazendo a Condição C1 e o resultado produzir seqüências (ou subseqüências) de pontos que satisfaçam as hipóteses do Teorema (2.7.1) são elementos comuns da convergência global dos algoritmos livres de derivadas propostos em [36] e [37], nos quais consideram-se a abordagem do padrão e da linha de busca, respectivamente.

Uma outra alternativa para a nuvem de pontos é a hipótese de caminhada aleatória para a otimização global, em particular para otimizar a função objetivo na qual usa intensivamente os recursos computacionais na evolução da busca. Na abordagem da função aleatória f assume-se que para se obter um caminho (ou amostra) de um processo estocástico *a priori*, é definido como a distribuição de probabilidade sobre uma classe de funções. Um processo estocástico implicitamente define várias outras distribuições de probabilidade tais como as distribuições dos valores das funções para qualquer $x \in S$, a distribuição do número de ótimos locais, a conjunta do tamanho, localização do valor da função do ótimo global. Deste modo, os valores da função são computados de forma seqüencial para diferentes pontos $x \in S$ podendo em princípio serem usados para cálculo da distribuição *a posteriori* sobre as propriedades de f. Esta informação *a posteriori* pode ser usada para responder questões de modo que a escolha do próximo ponto deverá trazer uma evolução de f para que o processo seqüencial possa ser terminado. A abordagem das funções randômicas (ou processos

estocásticos) para global otimização foi originada por Kushner, [38].

Deve-se antecipadamente esgotar a discussão sobre a aplicabilidade da abordagem do processo estocástico para o problema de otimização global resolvendo o seguinte dilema: de um lado, um processo estocástico *a priori* deve ser especificado o mais consistentemente com as propriedades de f tal como, continuidade e diferenciabilidade. Por outro lado, o processo deve ser matematicamente tratável, isto é, deve ser capaz de determinar as expressões explícitas das distribuições *a posterior*. Isto implica que exige um processamento intenso para se obter os pontos nos quais f evolui. Portanto, esta abordagem é apropriada para otimização global, porém paga-se o preço da lentidão na evolução da direção do ótimo.

Para definir o critério de convergência dos métodos estocásticos deve-se considerar:

$$\underset{x\in D}{\text{Max}} f(x) = \lim_{k\to\infty}\left[\frac{1}{k} log \int_D e^{kf(x)} dx\right] \tag{2.7.7}$$

Assume-se que o conjunto D de restrições é um fecho de um conjunto aberto limitado não vazio $f : D \to \mathbb{R}$ é uma função contínua. Um resultado segundo Hiriart-Urruty [39] que se pode obter é considerando uma função g continua, e estritamente positiva sobre D, a transformação de f em $\log g$ é transformada na seguinte expressão:

$$r_k(x^*) = \int_D \left[\frac{g(x)}{g(x^*)}\right]^k dx, \quad x^* \in D \tag{2.7.8}$$

onde x^* é um máximo global de g em D se, e somente se, a seqüência $\{r_k(x^*)\}$ for limitada.

Teorema 2.7.2 *Assumindo que D é um fecho de um conjunto aberto limitado não vazio e $f : D \to \mathbb{R}$ é contínua. Supondo-se ainda que f é globalmente maximizada*

em D *em um único ponto* x^**. Então a seqüência:*

$$x_k^* = \frac{\int_D x e^{kf(x)} dx}{\int_D e^{kf(x)} dx} \quad onde \quad k = 1, 2... \tag{2.7.9}$$

converge para x^* *quando* $k \to \infty$*.*

Apesar das diversas alternativas da condição de otimalidade persiste a dificuldade em estabelecer um critério que permita encontrar de forma eficiente a direção da busca. Considerando então, a inexistência de informação $\nabla f(x)$, neste caso seria impossível a minimização de uma função diferenciável, pois, uma condição necessária é o $\nabla f(x) \approx 0$. A alternativa para solução destes problemas é a utilização de métodos livres de derivadas ou da sua forma explícita. Neste caso, o preço que se paga é o esforço computacional e tempo de processamento.

Uma outra alternativa é fornecida pelas seguintes condições:

Dada uma função qualquer, num espaço euclidiano, não restrita (não necessariamente contínua e diferenciável) sobre um conjunto limitado não vazio (não necessariamente compacto), considerando a existência de pelo menos um ótimo na região de busca, existem seqüências de $\{f(x_k)\}$ que transformadas por uma função de estanciamento (função cotada superiormente) associada a um processo estocástico, fornecem seqüências de funções de distribuições de probabilidade, tais que, os momentos dessas distribuições irão fornecer para cada seqüência a direção em que o máximo mais provável se encontra em relação ao valor esperado daquela seqüência $\{x_k\}$.

São considerações: Seja f uma função qualquer (descontínua em p pontos) sobre S. Tomando uma seqüência de pontos (nuvem aleatória de pontos) sobre S, e admitindo-se que x_p seja um ponto de descontinuidade da função. Assume-se que o valor da probabilidade (valor que a função distribuição de probabilidade assume

no ponto) é nula, pois, para valores fora do intervalo [0,1] $P(x = x_p) = 0$. Logo o valor esperado de x_k^1 existe e a partir da sua distribuição se pode construir uma nova nuvem x_k^2 considerando o conhecimento *a priori* fornecido pelos momentos da distribuição anterior. Tomando-se em cada iteração $\text{Max}\, f(x_i^k)$ e substituindo o anterior somente se $\text{Max}\, f(x^k) < \text{Max}\, f(x^{k+1})$, então a seqüência $\text{Max}\, f^k$ é crescente e portanto, uma seqüência de funções monotonicamente crescente converge para um $\text{Max}\, f$ quando $k \to \infty$.

Com base neste Teorema pode-se perceber dois aspectos:

- O teorema garante a convergência do algoritmo para qualquer tipo de função sobre um intervalo qualquer.

- É idenficado um caminho de busca e atualizado para cada iteração evitando o direcionamento para um ponto de máximo ou mínimo local.

Os métodos baseados em processos estocásticos ou busca direta são conhecidos desde os anos 50 [40].

Capítulo 3

MÉTODOS DE BUSCA DIRETA

3.1 Introdução

Aspectos Relevantes

Considera-se o problema de encontrar o mínimo local de uma função $f(x)$ definida em $\mathbb{R}$. Se f é diferenciável e $\nabla f(x)$ pode ser calculado ou estimado por diferenças-finitas, um vasto número de métodos de otimização baseados em derivadas estão disponíveis para resolver este problema.

Supondo que a informação explícita sobre $\nabla f(x)$ está indisponível ou não é confiável. A tarefa de minimizar a função sem os recursos de derivadas deve parecer impossível desde que, para uma função diferenciável, a condição necessária para minimização é que $\nabla f(x) \approx 0$. A alternativa para solução deste problema consiste na utilização de métodos sem derivadas ou de busca direta.

Os métodos de otimização sem derivadas (ou livres de derivadas, de busca direta) são conhecidos desde os anos 50 [40]. Entretanto, pelos anos 70 estes métodos foram largamente disseminados pela comunidade matemática de otimização e em seguida

desapareceram da literatura por três razões básicas mais importantes:

- A direção da busca é desenvolvida por heurísticas ...
- Não há provas de convergência ....
- Muitas vezes a taxa de convergência é muito lenta.

Em 1991, o interesse nos métodos livres de derivadas foram retomados com a publicação de artigos no contexto de computação paralela. Estes itens vêm sendo fortemente reconsiderados observando a questão da convergência com mais detalhes, a partir de então.

O decisor ou o modelador típico deseja sempre encontrar a decisão ótima absoluta na qual corresponde a um mínimo (ou máximo) de uma função objetivo adequada, satisfazendo um dado conjunto de restrições viáveis. A função objetivo expressa especialmente a performance do sistema e desta maneira o lucro, a utilidade, a perda, o risco, ou o erro. As restrições são originadas por questões de natureza física, técnica, econômica — ou algumas outras considerações possíveis.

Um grande número de modelos de algoritmos de otimização tem sido desenvolvidos principalmente no contexto local tradicional, notadamente linear e programação convexa. Entretanto, existem também numerosos casos nos quais a estrutura necessária (linearidade e convexidade) não é satisfeita, ou não é de fácil verificação. O processo e o fenômeno podem ser de uma ordem qualquer (não-linear). As formas de funções mais freqüentes aplicadas em engenharia, física, química, economia, e estudos de meio ambiente são polinomiais, funções de potência, onde as exponênciais/logaritmicas e trigonométricas - aparecem na frente das descrições dos modelos de sistemas. Para modelagens aproximadas das funções polinomiais de baixa ordem

(freqüentemente, linear ou quadrática) produzem-se formas de modelos que são então gerenciáveis através da otimização tradicional relacionada em um ojetivo geral da natureza da não-linearidade — nos mais diversos contextos, conforme descritos em [22], [41], [42], [43],[44], [45], [46].

Se existir uma certa complexidade na descrição de um sistema não-linear, o modelo de decisão associado deve ter múltiplos ótimos locais. Nos casos mais realísticos o número de soluções locais não é conhecido *a priori*, e a qualidade entre a solução local e global deve diferir substancialmente. Conseqüentemente os modelos de decisão podem ser muito difíceis, e as regras de estratégias padrão não são diretamente aplicáveis para resolvê-los. Então, necessita-se de conceitos e técnicas de OG confiáveis e mais amplos que possam resolver um número maior de classes de problemas.

O tipo de contexto analisado neste trabalho envolve principalmente problemas não-lineares não restritos usando modelos de otimização sem derivadas. Muitas das propostas existentes neste contexto apresentam dificuldades em estabelecer a forma de tornar eficiente o processo de convergência [35] e [47]. Observa-se que muitas contribuições têm sido dadas devido à importância dos problemas de OG nos processos de decisão, principalmente no contexto de processamento paralelo a partir de 1991.

Na obtenção da solução global muitas dificuldades podem ser encontradas. Os algoritmos, atualmente utilizados, têm suas dificuldades particulares de implementação principalmente no que se refere a tempo de convergência e à precisão.

Pesquisas Prévias

Relevantes pesquisas são relacionadas a Otimização Global empregando algoritmos sem derivadas. Porém, visando dar suporte à discussão desta pesquisa, vários tópicos

conceituais da Otimização Global foram apresentados no Capítulo 2.

Para permitir estabelecer referências, formula-se o problema de OG como:

$$\text{Min } f(x) \tag{3.1.1}$$

$$\text{sujeito à } x \in C$$

onde $C = \{x : l \leq x \leq u; g_i(x) \leq 0 \quad i = 1, ..., I\}$ aplicando-se a seguinte notação:

- $x \in \mathbb{R}^n$: x é um n-vetor de variáveis de decisão.
- $f : \mathbb{R}^n \to \mathbb{R}$: é uma função objetivo contínua.
- $C \subset \mathbb{R}^n$: é um conjunto de decisão limitado não vazio (um subconjuto próprio de $\mathbb{R}^n$).
- C é definido por: l e u: explícitos, definem os limites inferior e superior de x.

  $g : \mathbb{R}^n \to \mathbb{R}^m$: é uma coleção finita de (I-vetores) funções de restrições contínuas.

A seguir apresenta-se uma lista básica de condições analíticas que garantem a existência de um conjunto não vazio de soluções ótimas se o problema for modelado nas condições acima, condição fornecida pelo Teorema Clássico de Weierstrass. Ao mesmo tempo, sem especificar as condições da estutura pode-se ter muita dificuldade. Por exemplo, C pode ser descontínuo, e uma grande quantidade de componentes pode não ser convexa, além disso, f pode ser multiextremos. Por isso, existe um número desconhecido de soluções ótimas globais e locais da equação (3.1.1). Por esta razão não existe uma caracterização algébrica geral da otimalidade global. Em contraste, na tradicional programação não linear existem muitos algoritmos exatos para resolver

o sistema de condições necessárias de Karush-Kuhn-Tucker: correspondendo a um sistema de equações e inequações.

A classe de modelos de otimização global contínua inclui um número elevado de modelos bem estruturados de casos específicos (por exemplo: minimização sobre restrições convexas). Na teoria, o que se deseja é encontrar todas as soluções do modelo dado pela equação(3.1.1), tacitamente assumindo-se que o conjunto de soluções é enumerável. O objetivo mais prático de OG é encontrar aproximações de um conjunto ótimo, denotado por X^* e o correspondente valor ótimo ($z^* = f(x^*)$ *para* $x^* \in X^*$). Então, necessita-se obter uma base de números em quantidade finita (nuvem de pontos) e infinitos números de passos de busca.

Existem várias maneiras de se classificar as estratégias de OG. Para a proposta deste trabalho é a divisão natural em métodos baseados em derivadas (determinísticos) e sem derivadas (heurísticos).

Os algoritmos determinísticos ou exatos usam o rigor teórico para encontrar a única ou todas soluções globais. Entretanto, o ônus para tornar de fácil aplicação computacional é alto para modelos com grandes dimensões, ou para modelos com funções complexas. Por esta razão, para grandes dimensões e sem um modelo especial estruturado, tem-se buscado solução de natureza prática nos métodos sem derivadas: algoritmos estocásticos e nos métodos heurísticos inteligentes que possuem uma componente de busca estocástica global.

Os métodos heurísticos, como regra, não têm estrita garantia da convergência, entretanto, em muitos casos, dão respostas rápidas, sendo uma alternativa na solução de modelos que não possuem uma solução teórica sobre metodologias rigorosas.

Quanto à questão da exatidão computacional depende a forma com que é colocada.

Por exemplo, a garantia de uma determinada precisão na aproximação de x^* em X^*, para uma ampla classe típica de modelos requerem um exponencial número de passos, com um modelo dimensionalmente crescente. Ao mesmo tempo, sobre condições suaves e sobre condições analíticas gerais, métodos estocásticos apropriados convergem para um elemento de X^* com probabilidade 1 (isto é, aproximadamente seguro). Com aplicações em um número razoável de tentativas, os métodos estocásticos têm uma probabilística garantia — ou seja, uma chance muito boa — de encontrar os pontos procurados que são soluções razoáveis e de boa qualidade.

Em função da importância do ponto, que nem sempre é suportado por rigorosas hipóteses de fato pode ser aplicada, tornando convincente a escolha da aplicação de heurísticas. Deste modo argumentos podem ser baseados na carência da implementação para um método exato apropriado ou, mais tipicamente na carência de um recurso computacional ou um tempo limite prático quando fornece aproximações de soluções com precisão aceitável. Entretanto, é essencial reconhecer também as limitações inerentes às hipóteses das heurísticas, e projetar performances somente com base em testes computacionais exaustivos.

Revisão da Literatura

A revisão é iniciada dando ênfase sobre as contribuições em métodos sem derivadas (seção 3.2) ou de busca direta, termo atribuído por Hooke e Jeeves em 1961, [14]. Este estudo é concentrado nos princípios da determinação do caminho da busca, que em muitos casos se utilizam de heurísticas. Algumas aplicações de OG são apresentadas observando a dificuldade do estabelecimento do caminho da busca e da convergência para o ótimo global.

Na seção 3.3 é apresentada uma discussão sobre métodos baseados em heurísticas. Em virtude da extensão das condições de convergência e da sua importância ainda no Capítulo 4 são avaliadas as estratégias da busca do caminho do ótimo e os seus critérios de convergência bem como os aspectos de precisão.

Na seção 3.4 é dada uma especial ênfase à avaliação e testes de *software* e fatores que afetam o desempenho.

Nesta revisão, é apresentado, no final de cada seção, um sumário sobre as características de cada modelo acompanhadas de comentários.

3.2 Métodos Estocásticos

Método adaptativo de busca estocástica

Procedimento baseado, parcialmente, em amostras aleatórias sobre C. Ajustes de estratégias, nuvens de amostras, opção de refinamento de soluções determinísticas, regras de parada, podem ser adicionadas como enriquecimento do esquema básico de amostras randômicas puras (que é uniforme sobre C). Estes métodos são aplicados em problemas discretos ou contínuos sob condições mais gerais [48],[49] e [50].

Algoritmos de Busca Bayesiana

Baseados sobre alguns postulados *a priori* usando modelos de funções estocásticas as quais fornecem a função f (ou um dado problema de OG) sobre determinadas condições. Uma subseqüente estimação adaptada das características de um problema-instância é baseada sobre o modelo. Tipicamente míopes (um passo ótimo, conseqüentmente fácil para computar) decisões aproximadas governam o procedi-

mento de busca. Métodos Bayesianos são apropriados para modelos gerais de OG [51].

Branch and Bound

Partições adaptadas, amostras e procedimentos de saltos (internos aos subconjuntos das restrições sobre o conjunto C) podem ser aplicados aos modelos de OG, analogamente aos problemas de programação inteira, ou um misto de metodologias inteira e linear. As hipóteses adotadas são utilizadas para muitos casos específicos, obtendo-se significantes generalizações. O método de *Branch and Bound* é aplicado em minimização côncava, diferenças convexas (DC) e problemas de otimização de Lipschitz. As referências aqui são: [52], [53], [54], [55], [56], [57], [58].

Estratégias Enumeráveis

Este método é baseado sobre a completa enumeração de todas as possíveis soluções. Aplica-se em problemas de otimização combinacional e para certos modelos de OG (tal como programação côncava) [54].

Métodos Homotópicos e de Trajetória

Esta estratégia tem o objetivo ambicioso de visitar (explicitar enumerando) todos os pontos estacionários da função objetivo f sobre o interior do conjunto C. Esta busca produz uma lista de todos os ótimos globais e locais. A metodologia é aplicável em problemas onde f é contínua. A carga computacional pode ser substancial [59] e [60].

Métodos Baseados em Integral

Nestes métodos o objetivo é a determinação de um supremo essencial da função f sobre o conjunto C, por uma aproximação dos conjuntos de níveis de f conforme descrito em [61] e [62].

Hipóteses simples (passivas)

Estes incluem, por instância, grades simultâneas de busca e busca randômica pura que são dadas pelo não interrelacionamento entre os pontos das amostras selecionadas (eles podem ser amostrados simultaneamente sem ser feita consideração alguma sobre um resultado individual). É intuitivamente perceptível que o método é convergente sobre condições analíticas suaves, sendo considerado como a esperança para resolver problemas de ordem e dimensão elevada [48] e [50].

Estratégia de Relaxação (aproximação externa)

Neste modelo a hipótese geral é baseada na substituição do problema por uma seqüência de subproblemas de fácil solução. Sucessivos refinamentos de subproblemas para aproximar do problema principal são aplicados: planos de corte e de cortes mais gerais, onde diversas funções minoritárias são construídas, e outras constumizações são permitidas. Algoritmos de relaxação são aplicados em diversas estruturas de problemas de OG - tais como minimização, ou programação com diferenças convexas [54] e [63].

3.3 Métodos Heurísticos

Aproximação Convexa Subestimada

Este método se utiliza da estratégia de tentativas para estimar as características da função objetivo se baseando em amostras diretas sobre o conjunto C. Muitas vezes, este procedimento funciona bem, porém em outros casos (quando o postulado dos modelos quadráticos não são ajustados) este procedimento não produz uma boa aproximação. Estratégias de subestimação são aplicadas em problemas de OG com funções objetivas contínuas [64].

Método Contínuo

Estes utilizam aproximações através da transformação da função objetivo em muitas funções contínuas com poucos locais minimizadores, e então se usa um procedimento de minimização local para tentar obter todos minimizadores de volta da função original. Métodos contínuos são aplicados em problemas de OG contínua [65].

Algoritmos Genéticos, Estratégias Evolutivas

Heurísticas de utilização de estratégias evolutivas baseadas em modelos de evolução de organismos biológicos. Vários algoritmos determinísticos e estocásticos podem ser construídos baseados em diversas regras de jogos evolutivos. Estas estratégias são aplicadas a problemas de natureza contínua ou discreta sobre requisitos e estruturas adequadas [11] e [66].

Extensão Globalizada de Busca Local

Esta estratégia prática está baseada em uma fase na busca global preliminar (grade passiva ou busca aleatória) acompanhada pelo escopo local da busca. Aplicável aos problemas contínuos de OG, a diferenciabilidade é usualmente postulada com a finalidade de adaptar a componente da busca local [48] e [50].

Melhoria Seqüencial do Ótimo Local

Estas aproximações incluem funções e métodos de tunelamento, esvaziamento e preenchimento. Tipicamente sobre funções auxiliares construídas e adaptadas para ajudar na busca gradual do ótimo (evitando os mais afastados). Estas estratégias são aplicadas a problemas de OG contínuos [67].

Recozimento Simulado - *Simulated annealing*

Esta técnica é baseada na analogia do resfriamento da estrutura cristalina para chegar a um estado de equilíbrio espontâneo caracterizado por mínimo global ou local de energia potencial. Recozimento simulado é aplicado em problemas discretos e contínuos sobre requistos de estruturas moderadas [68] e [66].

Busca Tabu

A idéia essencial é proibir nos movimentos da busca a visita aos pontos já visitados nos espaços de busca, ao menos nas próximas buscas. A metodologia da busca Tabu tem sido usada para resolver problemas combinacionais podendo ser extendida para problemas contínuos de OG [68], [66] e [11].

Pode-se observar que existem sobreposições de procedimentos entre os algoritmos

acima apresentados. Contudo, combinações de estratégias de busca são quase sempre desejadas e possíveis: isto conduz a projetos não-triviais de algoritmos.

Com esta breve revisão de subclasses de algoritmos de busca direta obtém-se informações básicas para realizar uma discussão detalhada sobre os aspectos da determinação da direção de busca e critérios de convergência. Não se pode deixar de avaliar as questões relativas às hipóteses necessárias para o funcionamento destes algoritmos. Neste sentido é realizada uma discussão sobre os métodos mais gerais no capítulo 4 pretendendo-se esgotar a discussão sobre estas questões.

3.4 *Software*

A respeito dos significativos avanços teóricos de OG, os padrões de uso ainda se encontram em desenvolvimento. Isto pode ser explicado por muitos fatores um dos quais está associado às dificuldades (numéricas e teóricas) da solução dos problemas de OG que são inerentes aos modelos.

Do ponto de vista puramente matemático, os problemas mais comuns de OG, por exemplo minimização côncava ou programação quadrática indefinida, fazem parte de uma classe de problemas de difícil solução. A dificuldade computacional de ter uma classe geral de problemas tem despertado interêsse em escala exponencial. As implicações práticas é que os problemas de OG definidos no $\mathbb{R}^n$, com $n = 5,\ 10,\ 20,\ 50,\ ou\ 100$ em uma regra geral levam a um crescimento muito rápido do esforço computacional. Conseqüentemente, se a capacidade de processamento do computador crescer também muito rápido, o custo do esforço da questão dimensional deve ser reavaliado.

Este fato conduz à uma conseqüência básica em relação a todas as possíveis implementações de metodologias de OG, de maneira que todos os algoritmos determinísticos, com suas regras, levam a um crescimento exponencial do esforço computacional. De um ponto de vista prático, poderá idealmente ser completado por uma correta e eficiente otimização na solução local. A convergência global, entretanto, pode ser garantida somente para as componentes de escopo global, ou seja, tomando-se o último ótimo e aplicando-se a busca exaustiva para verificação de um outro melhor.

A controvérsia entre rigor teórico e eficiência numérica é inevitável e mantém uma dificuldade essencial em desenvolver *softwares* eficientes e robustos. Uma outra forma é sacrificar a garantia teórica das propriedades de convergência determinística ou algumas eficiências numéricas, por exemplo, quando se resolve problemas práticos de tamanho e complexidade não triviais, sem se considerar um tempo realístico de trabalho (a convergência). Todo *software* tem endereço certo neste paradoxo, gerando a necessidade de avaliar o mérito do objetivo de forma cuidadosa, fazendo um balanço e avaliando a esperança prática contra o inteiramente correto.

Em formulações e soluções de modelos de decisões práticos um dos aspectos essenciais a considerar é a escolha do método e *software* para usar. O que se deseja é aplicar um método que seja o mais apropriado para resolver o problema dado.

O termo apropriado agrega diferentes critérios para diferentes decisores. Por exemplo, para o uso devem ser verificadas a confiabilidade e a velocidade do algoritmo para encontrar uma solução boa o suficiente sobre as mais diversas circunstâncias, mesmo que para a solução encontrada não seja garantida ser próxima do ótimo. Um outro decisor poderá dar mais atenção à rigorosa garantia e precisão da solução obtida, enquanto tem pouca atenção por eficiência.

Em um ambiente comercial, o *software* fornece nome e visibilidade no mercado, documentação confiável e suporte ao usuário devendo ser dominantes todos estes aspectos. Dependendo do critério do usuário, a escolha é quase sempre difícil.

Baseando-se também em exaustivas discussões na literatura sobre otimização, Khompatroporn *et al* 2001 [15], sumarizam as seguintes características evolutivas na qualidade de *software*:

- Generalidade. O algoritmo é insensível a detalhes secundários da estrutura do problema. Em outras palavras, o algoritmo teoricamente converge, sem sofrer nenhuma influência de restrições sobre a estrutura do problema (assumindo-se, é claro, que a classe do problema é apropriado ao escopo do algoritmo).

- Confiabilidade. O algoritmo é confiável e resolve o problema dado com um razoável grau de precisão, na qual pode ser especificado pelo usuário.

- Eficiência. O algoritmo guarda os cálculos em espaços tão pequenos o quanto possível e também, atualmente, o que importa na realidade é o tempo de processamento. Em geral a busca da solução requer uma certa quantidade de iterações devendo-se associar esta característica às outras.

- Fácil de usar. A aplicação do algoritmo deve ser relativamente fácil de usar e de se entender por usuários pouco experientes. O algoritmo deve ter poucos parâmetros que dependam do problema.

Portanto, percebe-se que *tradeoffs* entre atributos são inevitáveis. Notadamente, quando a robustez e a comodidade do uso aumentam, a eficiência tipicamente decresce e *vice versa*. Algoritmos que são muito sensíveis à seleção de ajuste de parâmetros são quase sempre problemas dependentes, e portanto não suficientemente gerais. Um

algoritmo ajustado para uma determinada classe de problemas com uma estrutura comum deve ser mais eficiente para solução de problemas daquela classe do que algoritmos que podem ser aplicados às outras classes [69] e [70].

3.4.1 *Software* Disponibilidade e Projetos

Em 1996 [56], uma pesquisa sobre *software* para otimização global foi apresentada sobre a forma de um relatório do *Mathematical Programming Society*. Esta pesquisa encontra-se disponível na *Web* por exemplo, *site* mantido por Mittelmann e Spelluci 2001 [71], baseado em respostas solicitadas aos autores dos *softwares* como também as informações coletadas na *Web* sobre 50 *softwares* produzidos. Neste momento a pesquisa mostra um número da ordem de algumas centenas de *softwares*. Por esta razão e também por conta da rápida mudança de cenário (novos produtos e implementações de versões, transição de versões grátis para comerciais, desaparecimento da *web* de *site* de *download*, entre outros) torna irracional a busca de detalhar uma lista específica de *softwares* produzidos e disponíveis para OG atualmente.

Em 2004, não existe um Guia *Softwares* de Otimização Global como apresentado por Moré e Wright em 1993 [21], certamente pela mudança rápida que vem acontencendo nesta área. Entretanto, existem muitos infomativos na *Web* voltados para este objetivo. Como referência alguns *Web Sites* são citados aqui:

- Progamação não linear questões mais freqüentes [13].
- Árvore de decisão para *software* de otimização global [71].
- A otimização global *site* de Neumaier [70].

Estes *sites* apresentam uma estrutura atualizada incluindo comentários e classifi-

cação dos modelos, algoritmos e *softwares* disponíveis e uma coleção de fatos e providências sobre informações relevantes. Especificamente os *sites* acima mencionados fornecem uma lista de *softwares* de otimização global, principalmente os publicamente disponíveis, alguns comerciais.

Como é de se esperar a qualidade do *software* varia largamente, nos termos do rigor teórico: nas condições de convergência precisa e nas características da qualidade (execução eficiente, confiabilidade, parametrização, uso-amigável, documentação, etc). Além disso, como certamente a maioria dos *softwares* livres disponíveis não apresentam suporte técnico e garantia de uso. Entretanto, para os desenvolvedores e usuários dispostos a investir tempo na sua evolução, as informações são encontradas podendo fazer bom uso delas e certamente aprender com as informações fornecidas.

A lista das características desejáveis, nos casos profissionais aplicáveis e indispensáveis aos produtos *softwares*, é a seguinte:

- Finalidade bem definida com plataforma de *hardware* e ambiente de *software*.
- Qualidade do guia do usuário manual do *software* e linha direta de ajuda, incluíndo um compreensivo esboço de modelo de desenvolvimento e processo de solução, modelagens sensíveis a erros com soluções, arquivos de usuários e exemplos numéricos não-triviais.
- Robustez funcional e interfaces amigáveis com o usuário.
- Rotina informativa de comunicação incluindo mensagens para todos programas de cenários de execução.
- Segurança em resolver procedimentos de seleção e de execução com opções de

entrada de parâmetros do usuário e mensagens de erro, além das próprias exceções (se necessário).

- Geração automática de arquivos com resultados que incluam todos as essenciais mensagens de rotina em adição ao resultado final.

- Visualização do modelo, sendo particularmente desejado em problemas de modelagem de sistemas não-lineares para dar assistência ao procedimento de desenvolvimento do modelo.

- Conectividade e flexibilidade para interoperabilidade com aplicações externas ao programa.

- Confiança e alta qualidade no suporte ao usuário.

- Manutenção e desenvolvimento contínuo do produto (desde plataforma de *hardware*, operação de sistema, adaptação aos ambientes com perceptívies mudanças).

Como os requisitos indicam, as tarefas não são simples e vão além do desenvolvimento de algum maquinário funcionalmente adequado para tratar números. Quando tenta-se adicionar à esta lista os critérios teóricos primários de alto nível (que são precisão, generalidade, confiabilidade e eficiência), então torna-se possível perceber o trabalho a ser desenvolvido.

Nos anos recentes, os desenvolvedores de modelos profissionais de ambientes voltados para otimização dedicam um considerável esforço para fornecer uma prova de quão amigável é o *software* produto. Exemplos destes trabalhos estão decumentados em relação ao AMPL ([72]), GAMS ([73]) e LINDO ([74]) e MPL (Maximal Software,

1998) [75] ou o de nível avançado Excel PSP Solver (Frontline Systems,2001) [76]. Em um nível similar de desenvolvimeto os *softwares* produtos têm um conjunto de padrões — por exemplo: para um OG profissional deseja-se ser reconhecido também como prático.

3.4.2 Teste de *Software*

A evolução dos algoritmos de otimização e *softwares* demandam dedicação, tempo e recursos de *hardware* em adicional objetividade. Estes comentários são particularmente válidos em relação aos *softwares* de otimização global, visto que OG envolve todos os modelos de programação matemática (PM). A definição de atributos que garantam a sua evolução é uma etapa difícil para o fabricante como também a definição do teste. O teste tem como um dos objetivos encontrar um realístico problema de OG que contenha tarefas não só no contexto atual como no de amanhã. Considerando estes aspectos constata-se na literatura a inexistência de um algoritmo que tenha a melhor *performance* em todas as categorias de problemas de OG, experimentos numéricos nos quais comparem novos algoritmos e *softwares* com os existentes e análise teórica completa do método de otimização. Por estes motivos realiza-se uma seleção de problemas de testes acompanhados por uma apresentação de sugestões como critério de evolução.

3.4.3 Fontes de Problemas de Teste

Em princípio, um modelo prático de otimização não linear deve servir como um teste válido para algoritmos apropriados. Entretanto, muitos modelos são também

específicos e complicados dificultando as suas reproduções ou, em alguns casos, seus detalhes são confidenciais.

Nestas circustâncias deve-se definir seu uso em um propósito geral para realizar testes comparativos. Conseqüentemente, o uso de um teste abrangente, reprodutível e transportável é também muito importante.

Existe uma considerável variedade de problemas de teste padrão. Estes problemas de teste são freqüentemente originados de aplicações reais, apesar de existir um número considerável de testes de natureza teórica propostos pela comunidade de OG.

Certamente é apropriado fazer um comentário sobre alguns tipos de testes teóricos (acadêmicos) que não devem ser usados, exclusivamente para provar a aplicabilidade de um determinado *software* ou algoritmo de OG. A figura 3.1 é um exemplo para ilustrar o tipo de função de teste. Note-se que este tipo de função é usada com o propósito do teste mencionado. Costuma-se também usar funções tipo ponto de sela onde a região na sua proximidade é quase convexa, com uma região muito extensa de platô (região de atração). Na proximidade do ponto de sela identifica-se não linearidade, porém o ponto chave colocado é que seja a solução através de algoritmos de busca global ou puramente algébricos não se devendo encontrar dificuldade alguma em obter a solução. Pelo menos nas últimas três décadas, um considerável esforço tem sido dedicado em colecionar e atualizar problemas de teste para otimização global não-linear. Para informação sobre problemas de programação não convexo e não-linear [77] ou [78].

Os dois volumes editados por Dixon e Szegö [79] antecipam o esforço para resolver modelos de OG: as contribuições também incluem um pouco das funções de teste clássicas. Floudas e Pardalos [80], Jansson e Knüppel [81], Floudas *et al* [57]

disponibilizam um extensivo conjunto de problemas de teste de OG.

Na *Web*, nas páginas de Fourer [13], Mittelman e Spellucci [82]encontram-se procedimentos de avaliação de teste de algoritmos e discussões sobre vários problemas de teste e sugestões sobre o relatório dos resultados de teste.

Uma classificação concisa de problemas de teste para OG é dado a seguir:

- Otimização não-linear irrestrita (quase sempre usa um ponto de partida).
- Otimização global não-linear sobre restrições gerais.
- Otimização global com restrição discreta.
- Problemas com estrutura especial exportável.

Em respeito à esta classificação, note-se que em otimização global discreta é necessária a garantia da existência da solução.

Note-se também que testes de otimização combinacional (OC) são considerados na estrutura de problemas de otimização combinacional global(POCG), desde que todos os modelos de OC possam ser equivalentemente formulados como POCG. O modelo de classificação conduz a um desafio por seu resultado em si.

Para estabelecer uma diretriz e *benchmarks*, as soluções aproximadas de OG e a implementação do *software* devem ser submetidos a um teste intensivo, sobre todos os tipos de modelos propostos pela comunidade de OG. Neumaier [70] coloca que um algoritmo razoável de OG deve se capaz de competir com algoritmos de solução de problemas não-lineares convexos, ou duplamente lineares sem um aumento significativo de tempo para obter a solução, comparado com o algoritmo de busca local específico. Este é um critério perfeitamente compreensivo que, entretanto, não é fácil de encontrar. Por exemplo, um algoritmo pode não atender a um propósito geral

de resolver problemas de OG contínua se comparado a um algoritmo próprio para o problema de programação linear (PL) de larga escala, a menos que também seja incluído um procedimento para reconhecer a estrutura especial de um problema de PL. Um caminho natural para se resolver simultaneamente várias classes específicas de OG é definir modelos de classes mais amplas, e assim definir estatégias para fazer uma pré-classificação da classe e modelo para submeter ao sistema solucionador em questão. Entretanto, isto deve oferecer uma alternativa extra para o usuário, não necessariamente bem vinda, um ônus quando submetido o modelo ao solucionador, ou isto requer esforço ao desenvolvimento de *software* para suportar um modelo de pré-classificação e pré-processamento automáticos.

Note-se também, que conseguir convergência global rigorosa e aceitável eficiência numérica são critérios parcialmente contraditórios. Na solução de problemas de OG o ótimo global é freqüentemente encontrado (que é uma aproximação) em uma fase inicial de busca, e uma boa parte do tempo computacional é gasto em verificar a (aproximação) otimalidade global da solução encontrada. As considerações sobre as regras de parada estão longe novamente de serem triviais, de maneira que em sua grande maioria são definidos sobre propriedades da classe do modelo da qual está associado o problema concreto. Métodos rigorosos de OG tipicamente gastam um esforço muito significativo sobre garantias (determinísticas) de fronteiras, enquanto estratégias heurísticas de OG, freqüentemente não se preocupam com estes aspectos. De acordo com a experiência da implementação prática, regras de estatísticas de decisão adequadas conduzem à aceitáveis eficiências e satisfatórias precisões, em muitos testes de problemas práticos. A apropriada implementação dos conceitos é, entretanto, pouco técnica, e quase sempre aumenta significativamente o esforço da

busca. Problemas de otimização global possuem inerentes dificuldades.

3.4.4 Evolução e Comparação dos Resultados de Teste

O objetivo experimental da análise comparativa de vários algoritmos de OG vem evoluindo o seu significado prático nas últimas décadas. Assim, é feita uma breve revisão sobre o estudo para comparação com *benchmark*obtendo-se maiores detalhes na fonte original Khompatroporn [15].

Colville [83] fez uma experiência pioneira para comparar a performance de algoritmos de otimização não-linear. Oito problemas de teste foram submetidos à 30 códigos. O modelo de teste foi baseado sobre problemas de motivação prática enviados por empresas que possuiam computadores de grande porte: IBM, Shell, Electricité de France, entre outros. Aos participantes foi solicitado para submeterem os resultados do seu melhor esforço e o tempo de execução para cada problema. Com este trabalho foi iniciada a busca do estabelecimento de critérios de avaliação. O estudo tinha sérias falhas, visto que as condições exatas de teste não tinham sido bem definidas.

Eason e Fenton [84] realizaram um estudo comparativo de algoritmos baseados na solução de 13 problemas de teste (mais tarde incluindo alguns problemas de Colville). O estudo inicialmente foca os métodos de penalidade e todos as performances computacionais sobre a mesma máquina. O estudo não inclui outros tipos de disponibilidade no tempo, e muitos modelos foram também fáceis de resolver.

O maior estudo computacional de algoritmos e *softwares* de programação não-linear foi realizado por Shittkowski [85]. O estudo inclui 20 diferentes códigos: isto foi ampliado para um conjunto de 180 problemas gerados randomicamente com características pré-determinadas e uso de pontos de multípla partida. Os códigos evoluem

baseados na sua eficiência, confiabilidade, convergência, habilidade para resolver degeneração, problemas de mau condicionamentos e fácil uso. Então um (obviamente subjetivo) esquema de pesos foi aplicado para obter um posicionamento final para cada algoritmo.

Nos anos recentes um número de testes de desafios e problemas reais foram propostos e em recentes artigos no *Journal of Global Optimization* e em outras partes.

Em suma, estudos computacionais para OG devem ser baseados sobre estes testes e similares. Especificamente, deve-se considerar medidas múltiplas de performance, uma compreensiva seleção de funções de teste e procedimentos bem definidos para realizar a avaliação.

3.4.5 Critério de Avaliação

Um relatório sobre um experimento computacional deve abordar aos seguintes aspectos:

- Modelo ou classe de problema (dimensão, número de restriçoes, região viável, tipos de parâmetros).
- Número de ótimos globais e locais (se conhecidos), valor ótimo ou melhor solução conhecida.
- Suporte prático ou outra razão para escolha (quando disponível).
- Relatório de trabalhos recentes (se disponíveis, para propósito de comparação).
- Soluções aproximadas usadas no teste: algoritmos e suas implementações, esboço conciso do projeto e código, parametrização do algoritmo e dados de saida.

- Plataforma de *hardware*, sistemas operacionais, e ambiente de *software* (compilador *software* adicionais) usados no teste.

- Metodologia do teste: descrição exata de todas as medidas de performances usadas (padronização do tempo de processamento baseado no modelo resolução em unidades de tempo, número de funções/gradiente/Hessiana de evolução, requisitos de precisão e outros critérios de parada, etc).

- Relatório dos quantitativos de sucessos e de falhas, com explicações adicionais.

- Análise do efeito dos parâmetros.

- Características estatísticas para o problema de classes randomizadas ou para métodos randomizados de solução.

- Sumário de resultados, em tabelas e gráficos (desejável).

- Comentários adicionais e recomendações.

Novamente, pode-se observar que os critérios apresentados freqüentemente apresentam conflitos parciais. Considerando-se, por exemplo, o *tradeoff* entre a precisão da solução e um (possível pré-conjunto) máximo número de funções de avaliação. Desta maneira, conceitos e técnicas usadas em otimização devem ser consideradas para avaliar suas robustez e fraquezas nos algoritmos de otimização.

3.4.6 Um Exemplo de Avaliação de *Software*

Usando um problema de teste randomizado, como exemplo, um conjunto de função de teste randomizada será usado para ilustrar o relatório computacional, Figura 3.1.

Especificamente, a função de teste randomizada proposta tem a seguinte forma geral:

$$f(x) = s\sum_{i=1}^{n}(x_i - x_i^*)^2 + \sum_{k=1}^{k_{max}} a_k sen^2[f_k P_k(x - x^*)] \tag{3.4.1}$$

Como a $f(x)$ é definido para um número fixo de n, tem como se segue:

- $s = 0,025n$ fator de escala que por definição, depende da dimensão do modelo.
- $l = -5$, $u = 5$ respectivamente, limite inferior e superior, idênticos para cada componente x.
- x^* solução randomicamente gerada, escolhida por uma distribuição uniforme sobre $[l, u]$.
- $a_k = f_k = 1(k = 1, ..., k_{max})$ escalas de amplitude e múltiplos de freqüência.
- $k_{max} = 2$ e portanto:

$$P_1(x - x^*) = \sum_{i=1}^{n}(x_i - x_i^*) + \sum_{i=1}^{n}(x_i - x_i^*)^2 \tag{3.4.2}$$

$$P_2(x - x^*) = \sum_{i=1}^{n}(x_i - x_i^*) \tag{3.4.3}$$

são termos de ruído polinomial.

Apesar da relativa simplicidade deste exemplo esta função é uma função de teste não trivial. O grande número de mínimos locais levam ao uso de uma busca global que pode ser observado na Figura 3.1

Critérios de Avaliação

- Número de ótimos globais 1, o valor correpondente à solução é zero.
- Número de ótimos locais: *a priori* não conhecido; cresce exponencialmente em função de n.

- A escolha foi motivada pela importância das funções trigonométricas que, em geral, modela problemas práticos e por uma dificuldade visível de instância de baixa dimensão.

- Trabalhos recentes disponíveis e não diretamente disponíveis.

- Solução de aproximação usada no teste: TALUS *software* apresentado neste trabalho Capítulo 6.

- Algoritmo usado: busca global através de nuvens probabilísticas.

- Estrutura dos dados de entrada: modelos de função são definidos e o usuário os seleciona através de menu de entrada. Veja Capítulo 6.

- Parametrização e critério de parada usado: valor de aceitabilidade de *threshold* 0,1. Tolerância de atender à condição de Kuhn-Tucker em 10^{-6}.

- Plataforma de *hardware* Pentium 500Mhz (computador pessoal).

- Sistema operacional : Windows2000.

- Ambiente de *software Dephi*.

- Tempos de processamento serão apresentados no Capítulo 6.

- Número de funções de teste é o fornecido acima.

- Relatório de sucessos e falhas: todo o teste foi executado com a precisão de 10^{-10}.

- Análise do efeito da parametrização são apresentados no Capítulo 6 .

As características estatísticas para a classe de problemas randomizados ou métodos randomizados de soluções randomizadas são apresentadas no Capítulo 6 onde são formuladas as comparações do problema abordado acima na solução do LGO (*An Integrated Application Development System for Continuous Global Optimization*).

3.4.7 Validação dos modelos descritivos do sistema

As definições bem estruturadas do modelo e do ambiente são condições essenciais para garantirem o sucesso de uma pesquisa. As principais fases da modelagem de sistemas quantitativos são bem diferenciadas e relatadas conforme a seguir:

- **Identificação**: formulação dos principais objetivos do modelo, determinação (seleção) da estrutura adequada do modelo.
- **Calibração**: Ajuste do modelo aos dados e resultados conhecidos (engenharia inversa).
- **Validação e aplicação em análise**: *forecasting*, controle e gerenciamento.

Conseqüentemente, a adequada ou melhor parametrização de um modelo descritivo é um importante estágio no processo de entendimento de um sistema complexo. As motivações da discussão sobre o modelo de calibração são apresentadas por [50], [86] e [87].

Uma forma razoavelmente simples e comumente aplicada a um modelo para calibração pode ser iniciada como a seguir:

- **Um modelo descritivo do sistema**: depende de certos parâmetros desconhecidos, seu vetor é denotado por x.

- **Conjunto viável** *a priori*: sobre o conjunto de busca S associado ao modelo.
- **Os valores de saída do modelo**: $\{y_t^m = y_t^m(x)\}$ nos momentos $t = 1, ..., k$.
- **Obter um conjunto de observações**:y_t para $t = 1, ..., k$.
- **Uma medida de discrepância**: definida como f expressando a distância entre os vetores $\{y_t^m\ \}$ e $\{y_t\}$.

O modelo do problema de calibração pode ser formulado como:

$$\text{Min } f(x) \tag{3.4.4}$$

$$\text{sujeito à } x \in C$$

onde $f(x) = f\left\{\{y_t^m(x)\}, \{y_t\}\right\}$. Tipicamente, C é um intervalo bem definido. Além disso, f é contínua ou uma função especial um tanto suave. Hipóteses sobre a estrutura de f são de difícil postulação. Para cada parâmetro do vetor x fixado, a seqüência de valores de saida $\{y_t^m(x)\}$ são produzidos por uma fórmula implícita, ou por um procedimento numérico computacional (como por exemplo: a solução de uma equação diferencial). Conseqüentemente, o modelo definido pela equação (3.4.4) representa um problema típico de OG pertencente à classe dos modelos contínuos. Entretanto, existe a necessidade de tornar o procedimento de OG uma solução do problema de calibração sobre as mais gerais condições apresentadas acima.

Para concluir a discussão acima é necessário introduzir o estudo de muitas variações sobre a possibilidade de calibração em detalhe para solução do problema declarado. No Capítulo 6 será apresentado como ajustar os parâmetros do TALUS na busca de uma boa precisão com um bom tempo de resposta.

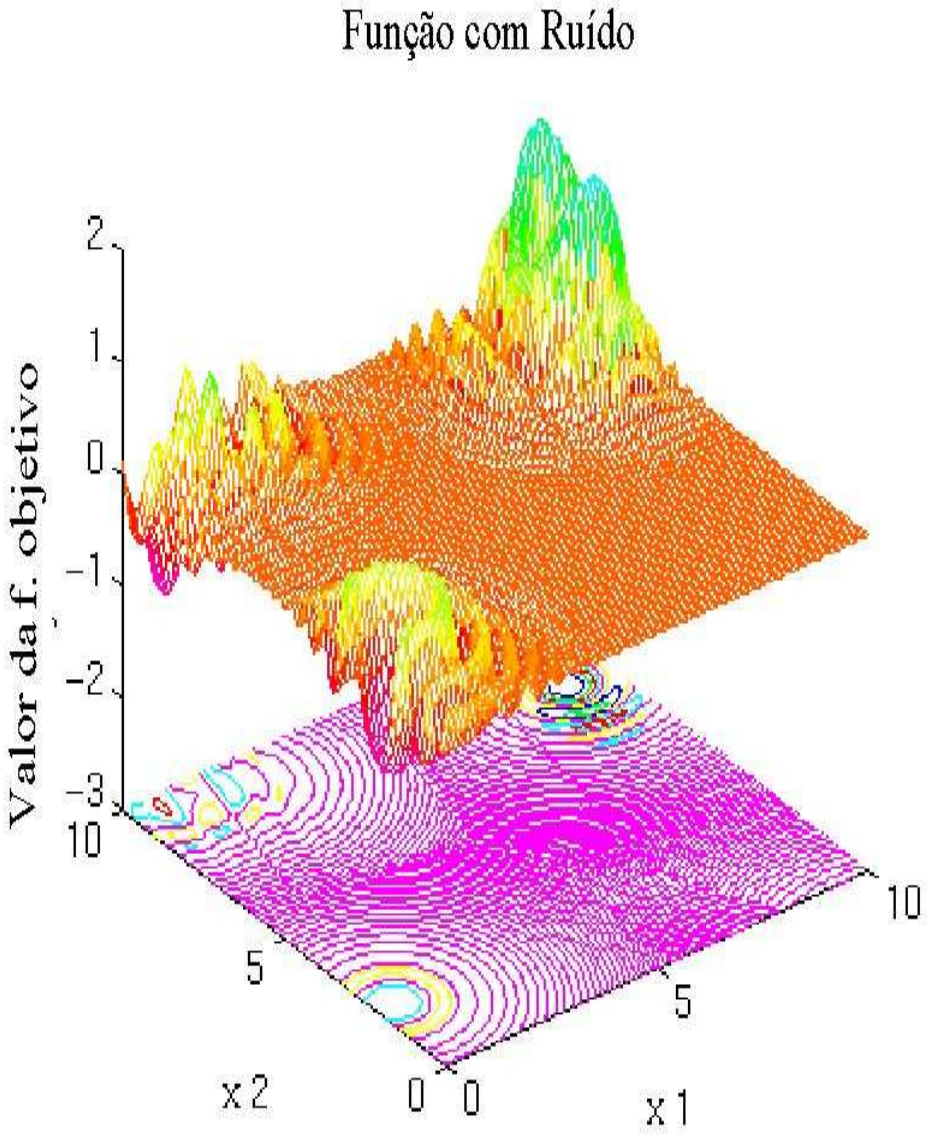

Figura 3.1: - Tipo de função de teste

Capítulo 4

SUPOSIÇÕES PARA CONSTRUÇÃO DE MODELOS DE BUSCA DIRETA

4.1 Bases para a Construção de Modelos de OG

Na seção 3.1 descreve-se alguns modelos particulares de busca direta para solução de problemas de OG encontrados na literatura com ênfase a identificar os critérios comuns de busca da direção mais eficiente do ótimo. Entretanto, é necessário o detalhamento de seus aspectos construtivos básicos que possam fornecer informações para a construção de modelos que possam ser mais eficientes. Devendo-se considerar os seguintes aspectos:

- Identificar os procedimentos de busca da solução ótima;
- Identificar os critérios de parada.

Muitos dos procedimentos usados para a construção de algoritmos de otimização podem ser encontrados em [35]. Entretanto, considerações específicas para identificar as dificuldades dos algoritmos atuais ainda são pouco abordadas. A busca de alternativas de modo a implementar alterações para atender um contexto específico ou mais geral tem sido objeto de pesquisas atuais. Neste capítulo discute-se com detalhes os princípios de funcionamento dos métodos de otimização global de busca direta.

Os procedimentos gerais para uso em modelos de otimização global, com o objetivo resolver uma larga faixa de classes de problemas de OG, são obtidos baseando-se nos modelos de busca direta através da análise dos procedimentos utilizados em algoritmos; como *Simulated annealing*, Genéticos, Tabu e Colônia de formigas. Então, no Capítulo 5 é apresentado o novo algoritmo de busca direta TALUS que agrega à sua estrutura propriedades que torna eficiente a estratégia de busca do ótimo global.

Observando-se os procedimentos disponíveis e considerando o contexto de OG, um procedimento básico é proposto para identificar as fases comuns aos diversos algoritmos de modo a estabelecer análises comparativas. Isto é apresentado sobre dois aspectos: o primeiro por macro-ações compostas por quatro fases, e o segundo dando para a terceira fase um maior detalhe composto por 10 passos. O macro procedimento é apresentado na Figura 4.1.

Fase 1	Definição da escolha da nuvem de valores iniciais para as variáveis

↓

Fase 2	Procedimentos para escolha do ótimo ou dos pontos com melhores resultados

↓

Fase 3	Determinação dos procedimentos de evolução da aproximação do ótimo através da escolha adequada da nova nuvem

↓

Fase 4	Critérios de parada e ajuste de precisão

Figura 4.1: Macro Ações de um Algoritmo de Busca Direta

A Fase 1 consiste em estabelecer a natureza do algoritmo (que classe de problemas ele tratará). A partir desta definição se estabelece como as funções de entrada serão capturadas pelo algoritmo. E desta forma se define uma seqüência de pontos, ou um conjunto de valores iniciais para as variáveis associadas ao problema a ser otimizado. Os elementos deste conjunto devem pertencer ao espaço de busca C.

Os procedimentos de escolha do critério de aproximação do ótimo, na Fase 2, (pontos que geraram os melhores resultados) dependem do algoritmo utilizado, pois, nele encontra-se a estratégia de quais pontos serão selecionados para fazer evoluir as próximas iterações. Na Fase 3 se estabelece a direção e o tamanho do passo através da estratégia de evolução adotada pelo algoritmo específico. Gera-se uma nova seqüência de pontos que, de princípio, promoverão a evolução de resultados. Este procedimento

está associado ao estabelecimento da direção e do tamanho do passo gerado para cada novo elemento da nova nuvem, função da nuvem anterior. Neste processo de alguma forma é incluído algum aspecto probabilístico às iterações, ou do processo de geração da nova nuvem.

Na Fase 4 são estabelecidos os critérios de convergência e ajuste de precisão. É neste momento que são ajustados os parâmetros de performance internos aos algoritmos. Estes parâmetros são usados conjuntamente com funções de teste, nas quais as soluções de OG são conhecidas de modo a permitirem os ajustes dos parâmetros. Desta forma, se pode ajustar os parâmetros de modo a obter os valores esperados com a precisão desejada.

Dado um problema de otimização global, deseja-se obter os máximos ou mínimos globais. Já foi apresentado a complexidade de se estabelecer algoritmos que atendam a uma faixa ampla de classes de problemas de OG, mas é possível se estabelecer procedimentos comuns entre os diversos algoritmos. Uma vez estabelecendo os aspectos comuns, é possível se identificar os critérios de escolhas das gerações de nuvens. Sendo assim, como cada algoritmo se comporta diante da incerteza da existência de algum ótimo em uma região qualquer do espaço $\mathbb{R}^n$, a fase 3 das macro ações deve ser detalhada de modo que se possa ter uma melhor compreensão dos procedimentos de decisão interna dos algoritmos. É claro, que os algoritmos dependem da existência prévia do ótimo, caso contrário, a busca ficará por tempo indefinido ou por algum critério de parada cujo resultado poderá não ter sentido.

1. **Processo de escolha dos pontos iniciais após substituição na função objetivo**: Consiste em determinar um conjunto de pontos melhor adaptados. O critério da escolha depende da estratégia usada em cada algoritmo.

2. **Estabelecer procedimentos de evolução dos pontos na direção do ótimo**: Determina-se aqui a estatégia de verificação do processo evolutivo das nuvens de pontos - cada algoritmo tem um particular procedimento.

3. **Definição da nova nuvem**: A definição da nova nuvem a ser usada na próxima iteração é determinante, pois, nela encontra-se associada a estratégia da determinação da direção de busca e o tamanho do passo a ser dado por cada elemento da nuvem.

4. **Avaliação de instabilidade ou de situações locais de estagnação da evolução**: Neste passo é avaliado se existe uma evolução a cada iteração ou se há oscilações em torno de um valor ótimo, ou se a nuvem convergiu localmente. Os critérios de avaliação são próprios de cada algoritmo.

PASSOS	ASPECTOS A SEREM DETERMINADOS
1	Definir procedimento para selecionar os pontos iniciais
↓	
2	Estabelecer procedimentos de evolução dos pontos na direção do ótimo
↓	
3	Definir a nova nuvem
↓	
4	Avaliar a instabilidade ou as situações locais de estagnação

Figura 4.2: Construção de um modelo de comparação funcional de algoritmos.

Como apresentado na Figura 4.1 os procedimentos, para avaliação do macro processo de funcionamento dos algoritmos de OG de uma forma geral, mostram as bases funcionais de construção dos algoritmos de OG, isto é como os algoritmos trabalham mesmo que em situações ou modelos de OG particulares. Neste contexto as fases apresentadas são simplificadas e os passos apresentados a seguir detalhados na Figura 4.2 fornecem um pouco mais de informação sobre a Fase 3.

Considerando estes aspectos destaca-se aqui a necessidade de se fazer uma melhor análise nos procedimentos adotados pelos principais algoritmos de busca direta utilizados atualmente. Os algoritmos mais citados na literatura atual são o *Simulated Annealing*, Genético, Colônia de formigas.

Desta forma, é realizada nas seções seguintes uma análise dos procedimentos destes algoritmos.

4.2 Principais Algoritmos de Busca Direta

Nesta seção são discutidos os aspectos construtivos de alguns algoritmos de busca direta mais citados na literatura. Para efeito de comapração com o modelo geral proposto na Seção 4.1 introduze-se a indicação da Fase correspondente ao modelo geral.

4.2.1 Algoritmo *Simulated Annealing*

Simulated Annealing usa a técnica de busca aleatória que evita aceitar um máximo local, isto é, introduzindo transições para valores crescentes ou decrescentes da função. Em cada iteração, procura o próximo candidato a ponto de máximo na vizi-

nhança do candidato corrente, agindo de acordo com a diferença entre os valores da função objetivo chamada, neste contexto, de função de energia ou função potencial. A principal vantagem deste algoritmo é evitar máximos locais: o algoritmo emprega, para isso, uma busca aleatória por vezes, aceita vizinhos cuja energia seja menor. Em algumas iterações, o *Simulated Annealing* tende a minimizar a função objetivo ao invés de maximizá-la.

Simulated Annealing explora a analogia entre o modo como um metal se resfria e congela e se estabiliza em uma estrutura cristalina de energia mínima, veja [16]. Entretanto, uma característica importante deste algoritmo é que a probabilidade de se aceitar um vizinho de menor energia decresce com o tempo, o que se implementa com um parâmetro, a temperatura, que decresce a cada iteração. Por fim, em qualquer temperatura, dados dois vizinhos de maior energia que o candidato a máximo corrente, A e B, energia(A) $>$ energia(B), a probabilidade de aceitação de A será maior que a de B.

O processo de minimização pode ser resumido no seguinte algoritmo:

Fase do Modelo geral	PASSOS	Procedimentos
1	1	Candidato $\geq S_0^*$
1	2	T $\leq$ T_0;
2	3	repita
2 2 2 2 3 4		próximo vizinho $\leftarrow$ vizinho do candidato tomado aleatoriamente; $\Delta E \leftarrow$ energia(próximo) - energia(candidato); se $\Delta \geq 0$ então candidato $\leftarrow$ próximo senão faça candidato $\leftarrow$ próximo com probabilidade $\exp(-\Delta E/T)$; T $\leftarrow$ próxima Temperatura (T);
4	4	$T < T_{final}$
4	5	Retorna candidato;

Figura 4.3: Algoritmo de Busca *Simulated Annealing*

onde:

- S_0 : estado (candidato a máximo) inicial;
- $\frac{T_0}{T_{final}}$;
- próxima temperatura T: função que calcula a temperatura vigente na próxima iteração;

Vale notar que T_0 , T_{final} e próxima Temperatura T são parâmetros de entrada adicionais do algoritmo, para os quais não há uma escolha única sempre eficaz.

Através de sucessivas iterações do *Simulated Annealing* em Bélisle [88] fornece a condição de absorção da deterioração da busca provocada pela busca do vizinho com energia menor na razão do resfriamento.

Teorema 4.2.1 (Bélisle, [88]) *Considere a seqüência* $\{X_n\}_{n=0}^{\infty}$ *gerada pelo Simulated Annealing. Faça a distribuição* $\mathbb{R}(x,.)$ *ser absolutamente contínua com função densidade* $r(x,.)$*, na qual é uniformemente limitada bem distante do zero. Além disso, assumindo que para muitos pontos de partida* x_0*, o programa de resfriamento converge para zero em probabilidade, a seqüência dos valores* $\{f(X_n)\}_{n=0}^{\infty}$ *converge em probabilidade para o máximo global* f^**, que é, para todo* $\epsilon > 0$ *e* $x_0 \in C$*,*

$$\Pr[f(X_n) < f^* - \epsilon \mid X_0 = x_0] \to 0 \text{ com } n \to \infty \tag{4.2.1}$$

Na prática guarda-se o valor corrente de f da seguinte forma:

$$f_n^* = \max_{0 \le k \le n} f(X_k) \tag{4.2.2}$$

4.2.2 Colônia de Formigas

O algoritmo Colônia de Formigas [89] (ou Algoritmo das Formigas) tem sido usado em muitas aplicações para solução de problemas de OG. O algoritmo é basicamente um sistema com multi-agentes (isto é, formigas artificiais) resultando em um complexo comportamento de uma colônia de formigas. O algoritmo de otimização colônia de Formigas é similar ao método de busca aleatória ou método de difusão, porém acrescido de interação entre os andarilhos para encontrar resultados mais rapidamente.

Uma formiga é um agente que se movimenta em todo o espaço de soluções do problema dado. O movimento é basicamente randômico, porém com um adicional. Quando uma formiga encontra uma boa solução para o problema ela libera um cheiro ao longo do caminho até a solução. Este cheiro é então percebido por outras formigas que aumentam as suas probabilidades de moverem-se na direção desta solução com cheiro forte. Este processo é similar ao usado por formigas reais para comunicar a fonte de alimento; daí o nome "formiga" para os agentes. Formiga é apropriado para problemas que tem uma representação natural em um espaço métrico de baixa dimensão. O algoritmo de otimização Colônia de Formigas é inspirado em colônia reais de formigas. Na natureza as formigas se comunicam por depositar feromonas (substância com cheiro ativo) para indicar o caminho entre uma restrição e o alimento.

Note-se que o caminho usado pelas formigas é uma forma diferente da solução tradicional de problemas de OG. O cheiro é uma habilidade das formigas resolverem problemas de otimização. Estes cheiro determina o caminho de aproximação. A forma de liberação do cheiro segue um critério de não-linearidade. O objetivo é obter uma forma de interação mais explícita entre os agentes.

O algoritmo é iniciado por N formigas uniformemente distribuídas na região de

busca. Cada formiga escolhe aleatoriamente um vizinho da sua atual posição e move-se para ele, com probabilidade não uniforme sobre a escolha dos vizinhos. É mais provável a escolha de uma melhor posição do que para uma posição de menor significância. As probabilidades de ocupação de um novo ponto depende da existência do cheiro. Inicialmente, no terreno não existe cheiro. Tão rapidamente uma formiga encontre uma posição mais favorável para encontrar alimento ela interrompe a busca e libera o cheiro ao longo do caminho do objetivo. A performance da interação entre as formigas depende da caminhada aleatória com probabilidades que são dependentes de situações de posições anteriormente ocupadas.

Ocupando uma posição a formiga x usa três tipos de informações para determinar a futura posição:

- A primeira diz respeito à informação geográfica $T(y_j, x_i)$ onde $j = 1, ..., m-1$ e $i = 1, ..., m-1$ que simplesmente fornece quão longe são as posições dos vizinhos de y_j e x_i. Esta informação é pré-determinada e iniciada com a criação desta base de dados sobre o espaço de busca.

- A segunda componente, denotada por $F(y_j)$, é relacionada com a importância estratégica das diferentes localizações y_j, isto é, o melhor ponto que as formigas podem encontrar. O operador humano pode alterar este parâmetro usando a sua experiência indicando as partes importantes da região pesquisada.

- A terceira parte é a distribuição do cheiro por outra formiga, $S(y_j, t)$. Para $t = 0$, que é inicializado para 0 e todo y_j:

$$S(y_j, 0) = 0 \tag{4.2.3}$$

O cheiro S deve ser iniciado durante o processamento do algoritmo conforme a seguinte equação:

$$S(x_j, t+1) = S(x_j, t) + \frac{j}{M} \tag{4.2.4}$$

A probabilidade de transição de ir para o local y partindo do local x no tempo t é dado por:

$$p(y_j, x_i, t) = 0 \tag{4.2.5}$$

se x e y não são vizinhos próximos, e:

$$p(y_j, x_i, t) \ \alpha \ \frac{1}{T(y_j, x_i)} + w_s S(y_j, t) + w_f F(y_j) \tag{4.2.6}$$

caso contrário.

Na equação 4.2.6 , $T(y_j, x_i)$ é a informação geográfica relativa ao tempo necessário para uma formiga mover-se de x_i para y_j, S e F são o cheiro e o valor do campo definidos acima, e $w_s = 1$ e $w_f = 1$ são pesos determinando a importância relativa de diferentes campos. A constante de proporcionalidade é determinada para atender ao requisito de que a soma de todos os y_j é igual a 1.

Um possível valor para o campo F é incluído em uma informação para a faixa de visibilidade do alimento para diferentes pontos. A desvantagem de fazer isto é que adiciona uma armazenagem ou requisitos de computação para o conhecimento do campo, isto destrói os atrativos da simplicidade do modelo das formigas.

Para evitar *loops*, é adicionada uma pequena tendência para o retorno de onde a formiga se originou. Se a nova posição é igual à antiga posição da formiga, gera-se um novo número aleatório e a nova posição será obtida. Isto reduz a probabilidade de uma formiga retornar em seu próprio rastro. Todas as formigas vão avançando paralelamente no tempo. Tão rápido quanto possível uma formiga atinja um alvo

local é então liberado um traço de cheiro. O cheiro é distribuído ao longo do caminho que levou ao alvo. O caminho que a formiga viajou é dado por uma seqüência de x_i para $i = 0, 1...M$ (com x_0 igual à posição de partida). O cheiro é também usado por outras formigas para determinar no tempo futuro a sua evolução, veja Figura 4.4.

A medida da velocidade de convergência das formigas é calculada pela distância de

Fase do modelo geral	PASSOS		Procedimentos
1	1		Enquanto o tempo máximo não termina
1		(a)	para toda formiga i:
1		i.	o conjunto x_i = posição corrente da formiga i.
2		ii.	aleatoriamente seleciona-se um vizinho de x_i usando a equação 4.2.6 e move-se a formiga i para lá ;
3		iii.	se a formiga i atingir ao alvo então:
3			A. liberar o cheiro em todos os sites visitados pela formiga i de acordo com a equação 4.2.4.
3			B. eliminar a formiga i.
4		(b)	se $S(x_i)$ não tem mudado , sair do *loop*
4	2		liberar cheiro com potencial efetivo

Figura 4.4: Pseudo código para algoritmo das formigas

Kullback-Leibler em [90]. É uma medida para avaliação da entropia relativa a duas variáveis do mesmo tipo, caracterizadas por suas distribuições de probabilidade f e $f^{'}$. A entropia relativa é dada por:

$$K(f||f^{'}) = \sum_{i=1}^{m} f_i \times \ln \frac{f_i}{f_i^{'}}$$

Observa-se que esta medida é assimétrica.

Adaptando-se estas equações ao problema obtém-se:

$$K(f, g; t) = \sum_{x} g(x, t) \ln \frac{g(x, t)}{f(x)} \tag{4.2.7}$$

$$f(x) = \frac{U_0(x)}{\sum_{y} U_0(y)} \tag{4.2.8}$$

e

$$g(x,t) = \frac{S(x,t)}{\sum_{y} S(y,t)} \tag{4.2.9}$$

A distância de Kullback decresce exponencialmente com o tempo. Este algoritmo não necessita de postulado algum; apenas requer a localização de alvos. O parâmetro que governa a distribuição das formigas no espaço de busca é o cheiro, pois, é o cheiro que vai fazer as formigas convergirem para um determinado alvo.

4.2.3 Algoritmos Genéticos

Um algoritmo Genético é um mecanismo de busca baseado na teoria da evolução das espécies. Tendo sido estabelecido como uma aproximação válida para problemas cujo requisito é a busca eficiente, os algoritmos genéticos têm sido aplicados de maneira crescente para solução de problemas de OG nas diversas áreas do conhecimento.

São baseados na teoria da evolução das espécies, lançada pelo fisiologista inglês Charles Darwin em seu livro "*The Origin of Species*" em 1858. Em 1975 por John Holland e seus alunos foram os primeiros a aplicarem o algoritmo genético em um ambiente computacional, e só mais tarde foi popularizado por David Goldberg, em 1989 [91].

Este algoritmo é computacionalmente simples e possui uma potente forma de evoluir a busca. Trabalha em um amplo espaço de busca discreto e necessita de poucas suposições sobre este espaço. Entretanto, por causa desta versatilidade e da pouca base matemática necessária à sua aplicação tem sido bastante aplicado mesmo que sendo questionável a sua performance.

A composição dos algoritmos Genéticos compreende três mecanismos básicos: reprodução, interseção (*crossover*) e mutação. Tipicamente o espaço de busca discreto pode ser varrido por seqüências de códigos como um conjunto binário de caracteres de comprimeto γ. Estas seqüências são análogas aos cromossomos. Uma população inicial de n seqüências é escolhida. Para simplificar n é escolhido par. O objetivo do algoritmo genético é encontrar o máximo da mesma função objetivo f (chamada de função de aptidão) definida sobre o espaço de busca.

Iniciando com uma população original, uma nova população de n seqüências é selecionada em três estágios:

- Primeiro, um conjunto de n seqüências (não necessariamente distintas) é escolhido na população original. Para selecionar cada membro da população com probabilidade proporcional à sua adaptação, estes indivíduos são selecionados para gerarem uma nova população, este processo é chamado reprodução.

- Segundo, o novo conjunto de seqüência é dividido em pares de forma aleatória. Para cada par ocorre o cruzamento da metade da seqüência que forma o seu código, desta forma gerando dois novos indivíduos cada um com uma parte do código de cada gerador, o cruzamento ocorre com probabilidade χ. Se o cruzamento ocorrer, então a posição entre 1 e $\gamma - 1$ é uma escolha aleatória, e a primeira metade da primeira seqüência no par é uma combinação com a segunda metade da segunda seqüência, a segunda metade da segunda seqüência é uma combinação com a segunda metade da primeira seqüência para gerar um novo par de seqüências (como os cromossomos).

- Terceiro, cada bit de cada seqüência na nova população é alterado (muda de

zero para um ou vice versa) com probabilidade μ. Isto é chamado de mutação uniforme.

Usualmente a probabilidade de cruzamento χ e a probabilidade de mutação μ são pequenas. O valor típico para μ é menor que 0,05. Algoritmos genéticos mais complexos têm taxa de cruzamento não uniforme entre iterações, ou taxa de mutação não uniforme entre iterações, ou ambas. Outras possibilidades incluem esquemas de cruzamentos e mutações mais complexos (exemplos: múltiplos pontos de cruzamento, mutação de blocos de dígitos simultaneamente).

O processo de reprodução probabilisticamente assegura que indivíduos de alta adaptação são perfeitamente mantidos na população, enquanto o cruzamento e mutação são necessárias para introduzir a diversidade necessária para garantir que o espaço de amostra inteira é atingível e evita emperrar numa solução subótima. A mutação é em si suficiente para introduzir diversidade. A idéia é que após um grande número de iterações a população consista principalmente de indivíduos muito próximos da solução ótima. Quando o algoritmo genético é finalizado, a adaptação de cada elemento da população passou por processo de evolução e o membro correspondente à máxima adaptação é tomado como o ótimo do espaço de busca.

Entretanto, com este método é possível que uma ótima solução ser obtida e ser perdida ao longo do processo de seleção, bastando para isso ocorrer algumas mutações e cruzamentos, pois, nem sempre um indivíduos mais bem adaptados sejam solecionados para os cruzamentos ou sejam preservados no processo de mutação. Um método mais efetivo, no qual requer um esforço adicional, é comumente usado: salvar em cada estágio o membro da população melhor ajustado e atualizar em cada iteração, isto é chamado de “a abelha rainha"ou “caracter elitista", veja Figura 4.5

Fases do modelo geral	PASSOS	Procedimentos
1	1	Geração da população inicial
2		Avaliação de cada indivíduo da população
2	3	Aperfeiçoar os indivíduos da população atual até que o critério de parada seja satisfeito
2 3 3 4		a- Seleção dos indivíduos mais aptos ; b- Criação de novos indivíduos a partir da população inicial usando operadores genéticos; c- Armazenamento dos novos indivíduos em uma nova população; d- Avaliação de cada indivíduo da nova população na procura de soluções satisfatórias.

Figura 4.5: Pseudo código para algoritmo Genético

Heuristicamente os algoritmos genéticos na prática apresentam bons resultados, porém, pouco fundamento matemático justifica sua performance. Um dos problemas chaves é decidir quando parar. Uma abordagem simplista é parar após um (grande) número de iterações fixado. A abordagem mais sofisticada foi feita por Jong(1975) [92], onde é definida a performance *on-line* como a média de todas as evoluções da função de adaptação incluíndo o valor corrente. A performance *off-line* (descreve convergência) é a média corrente da melhor performance para uma iteração particular. A mais sofisticada alternativa para parada após um fixado número de iterações é a parada depois de um ou de outro *on-line* ou, preferencialmente *off-line* após estabilizar.

Como os algoritmos genéticos são de busca estocástica, isto é, nunca é possível garantir que uma solução ótima tenha sido encontrada após um número fixo de iterações, e existe sempre a probabilidade de não ser encontrada a solução ótima. Entretanto, deve-se examinar a classe de convergência em probabilidade: dada uma probabilidade fixa δ $(0 \leq \delta \leq 1)$ pergunta-se qual é o menor número de iterações requerido para

garantir que se tenha obtido a solução ótima com probabilidade δ. Denota-se este número por $t(\delta)$.

Pesquisas têm sido desenvolvidas para definir a faixa de $t(\delta)$ para modelar algoritmos genéticos como cadeias de Markov. Em [93] demonstra como os algoritmos genéticos apresentam um eqüilíbrio pontual, muitas vezes aparece convergência para um falso subótimo após um longo tempo relativo ou movendo-se sobre a melhor solução. Este comportamento é muitas vezes verificado na prática. Deste modo, o critério de convergência apresentado *on-line* ou *off-line* pode ser enganoso. Em [94] caracteriza-se o algoritmo genético como cadeia de Markov. É mostrado que se o estado da variável é tomado por uma população corrente, então a cadeia de Markov é ergódica se, e somente se, a probabilidade de mutação for estritamente positiva. Então um comportamento de estado estável existe e não é possível para a cadeia de Markov permanecer em um estado subótimo indefinidamente.

Em [95] usa esta formulação da cadeia de Markov para encontrar faixas de $t(\delta)$. Para $x \geq 0$ define INT[x] para ser o menor inteiro tão grande ou igual a x. É demonstrado que $t_2(\delta) \leq t(\delta) \leq t_1(\delta)$ onde :

Teorema 4.2.2 (Convergência de algoritmos genéticos)

$$t(\delta) \leq \widetilde{t}_1(\delta) = \text{INT}\left[\frac{\ln(1-\delta)}{n \ln\left[1 - \min\left\{(1-\mu)^{\gamma-1}\left(\frac{\mu}{K-1}\right), \left(\frac{\mu}{K-1}\right)^{\gamma}\right\}\right]}\right] \tag{4.2.10}$$

onde K é a quantidade de seqüências de comprimento δ e $1 - \mu \leq \frac{K-1}{K}$, veja [96].

Uma aproximação prática é dada pela seguinte expressão:

$$t_1(\delta) \approx -\frac{(K-1)^{\delta n} \ln(1-\gamma)}{\mu^{\delta n}} \tag{4.2.11}$$

se $t_1(\delta)$ for grande, a sua faixa pode ser dada por:

$$\widetilde{t}_1(\delta) \approx -\frac{(K-1)^{\delta}\ln(1-\gamma)}{n\mu^{\delta}} \tag{4.2.12}$$

Considerando $\widetilde{t}_1(\delta)$ grande. O raio é portanto:

$$\frac{t_1(\delta)}{\widetilde{t}_1(\delta)} \approx n\left(\frac{\mu}{K-1}\right)^{-\delta(n-1)} \tag{4.2.13}$$

O que se pode perceber é que o número de iterações para garantir a convergência com uma certa probabilidade depende principalmente do comprimento do código que representa cada indivíduo e do tamanho da população.

De uma forma geral pode-se afirmar que o diferencial entre os diversos algoritmos de busca direta está na escolha dos elementos para formar as nuvens, embora que a nuvem inicial seja puramente aleatória. Além disso observa-se o critério que determina a evolução da busca, este sim é o critério de maior importância que combinado com a escolha da nuvem define quão rápido o algoritmo irá convergir para o ótimo global. Deve-se observar também a preocupação em tornar sempre simples o algoritmo para ser fácil a sua aplicação.

Capítulo 5

CONSTRUÇÃO DO NOVO ALGORITMO DE OG - TALUS

5.1 Introdução

No Capítulo 4 foram colecionados alguns aspectos que suportam os algoritmos de busca direta. Embora, de aplicação simples, estes algoritmos não possuem garantia de convergência para um número de iterações fixado. Entretanto, apresentam aspectos interessantes como adaptabilidade à uma grande gama de classes de problemas de otimização global.

Este capítulo é dedicado à construção do algoritmo TALUS. É uma proposta para melhorar a eficiência na determinação da direção da busca do ótimo global, incluíndo suposições diferenciadas sobre o comportamento da nuvem de indivíduos (pontos) utilizados em cada iteração.

5.2 Formulação dos Pressupostos Básicos

5.2.1 Descrição do Problema da Direção da Busca

É sabido que, quando o gradiente está disponível, para definir a convergência para ótimos locais de um algoritmo, para problemas sem restrição não se tem dificuldade. De fato, para cada iteração, o gradiente fornece informações para:

- Calcular e selecionar uma boa direção de decaimento na qual a função objetivo decresce com taxa apropriada.

- Determinar um comprimento de passo suficientemente grande ao longo da direção de decrescimento, chamado de "comprimento do passo" na qual é capaz de explorar apropriadamente o decrescimento na direção da busca, por forçar um significante decrescimento no valor da função objetivo relacionado com a norma do gradiente.

Quando o gradiente não está disponível, perde-se a informação a cerca do comportamento local da função objetivo. De fato, a $i-$ésima componente, $\nabla_i f$, do gradiente é uma derivada direcional da função ao longo do vetor e^i, e $-\nabla_i f$ é a derivada direcional ao longo do vetor $-e^i$. Então, todo o vetor gradiente fornece a taxa de mudança da função objetivo ao longo de 2^n direções $[e^1, e^2, ...e^n, -e^1, -e^2, ... - e^n]$. Este fato garante que a informação do gradiente caracteriza uma precisão total no comportamento local da função objetivo na vizinhança do ponto na qual as derivadas são calculadas.

Como foi visto no Capítulo 4 os métodos de busca direta têm estratégias diferentes, porém com o mesmo objetivo de sobrepujar a falta da informação contida no

gradiente. A abordagem comum está baseada na idéia de investigar o comportamento da função objetivo na vizinhança de um ponto genérico por amostragem da função objetivo ao longo de um conjunto de direções. Certamente que cada um destes algoritmos apresentam propriedades e características nas quais dependem de uma escolha particular do conjunto de direções com que são tomadas amostras da função objetivo.

As direções a serem usadas em algoritmos de busca direta poderão ser baseadas em cada comportamento local da função objetivo ao longo de uma indicação suficiente desse comportamento na vizinhança de um ponto. Esta direção deve ter a propriedade que, em cada amostra haverá um refinamento possibilitando:

1. Obter a indicação de que o ponto corrente é uma boa aproximação de um ponto estacionário da função objetivo, ou

2. Encontrar uma direção específica ao longo da qual a função objetivo decresce.

O ponto importante é identificar um grande número de conjuntos de classes de direções de busca na qual podem ser usadas para definir a convergência global para algoritmos de busca direta. No final da próxima seção é apresentada a proposição de uma condição geral na qual formaliza as classes de conjuntos que são aderentes às propriedades (1) e (2).

Em adição à contribuição para os pontos prévios (1) e (2), o método de amostragem dirigida tem a incumbência de guiar a escolha de um novo ponto, na qual a seqüência de novos pontos com segurança produzirão para o algoritmo uma convergência global para um ponto estacionário da função objetivo. Apoiando-se nas características comuns das técnicas de amostragem dos métodos da direção de busca propostos em [36], [37] e [97], na seção 5.2.2 são definidas as condições sobre amos-

tras da função objetivo ao longo da direção da busca apropriada para a convergência global dos métodos de busca direta.

5.2.2 Direções de Busca

Antes de descrever e analisar o modo de obter a direção da busca é necessário realizar algumas suposições sobre a função objetivo.

$$\text{A função } f : \mathbb{R}^n \to \mathbb{R} \text{ é contínua e diferenciável} \tag{5.2.1}$$

O primeiro passo para definir o método para obter a direção da busca é associar um conjunto apropriado de direções $p_k^i,\ i = 1, ..., r$, com cada ponto x_k produzido pelo algoritmo. Este conjunto de direções tem a propriedade de um comportamento local da função objetivo que ao longo delas fornece informação suficiente para superar a falta da informação do gradiente.

Aqui se introduz a nova condição na qual caracteriza o conjunto de direções $p_k^i,\ i = 1, 2..., r$, que satisfaz esta propriedade. Esta condição requer que a distância entre os pontos gerados por um algoritmo e o seu conjunto de pontos estacionários tenda a zero se e somente se as derivadas direcionais $p_k^i,\ i = 1, 2..., r$, assumam valores não negativos. Formalmente tem-se a seguinte condição: dada uma seqüência de pontos $\{x_k\}$ e a seqüência de direções $\{p_k^i\},\ i = 1, ..., r$, são limitadas tal que:

$$\lim_{k\to\infty} \|\nabla f(x_k)\| = 0 \ \Leftrightarrow \ \lim_{k\to\infty} \sum_{i=1}^{r} \min\{0, \nabla f(x_k)^T p_k^i\} = 0 \tag{5.2.2}$$

Teorema 5.2.1 (Convergência geral dos algoritmos de busca direta) *Conforme [35] faça $\{x_k\}$ ser uma seqüência limitada de pontos e faça $\{p_k^i\},\ i = 1, ..., r$, ser seqüências de direções as quais satisfazem à equação(5.2.2). Para cada $\eta > 0$, existe*

$\delta > 0$ tal que, para todo, porém finito, k, se x_k satisfaz $\|\nabla f(x_k)\| \geq \eta$, então existe uma direção $p_k^{i_k}$, com $i_k \in \{1, ..., r\}$, para o qual:

$$f(x_k + \alpha p_k^{i_k}) \leq f(x_k) - \gamma\alpha\|\nabla f(x_k)\| \ \|p_k^{i_k}\| \tag{5.2.3}$$

para todo $\alpha \in (0,\ \delta]$.

O teorema acima garante que, sempre que um ponto corrente não for um ponto estacionário, é possível forçar um decrescimento suficiente da função objetivo pelo uso de um dos conjuntos de direções que satisfazem a equação(5.2.2).

Do ponto de vista teórico, o Teorema(5.2.2), mostra que a condição fornecida pela equação (5.2.2) é um requisito suficiente para cada direção assegurar a convergência global da seqüência de iterações (pelo menos uma subseqüência) para um ponto estacionário de f. De fato, a condição fornecida pela equação (5.2.2) pode ser considerada tecnicamente aderente aos conjuntos de direções de busca nas quais podem ser satisfeitas ou facilmente forçadas em algoritmos de busca direta. Veja figura (5.1).

PASSOS		Procedimentos
1	Dados	$x_0 \in \mathbb{R}^n,\ \widetilde{\alpha}_0 > 0,\ \gamma > 0\ \theta \in (0,1)$
2		Faça $k = 0$
3		se existe $y_k \in \mathbb{R}^n$ tal que $f(y_k) \leq f(x_k) - \gamma\widetilde{\alpha}_k$, então vá para o passo 6
4		Se existe $i \in \{1, ..., r\}$ e um $\alpha_k \geq \widetilde{\alpha}_k$ tal que $f(x_k + \alpha_k p_k^i) \leq f(x_k) - \gamma(\alpha_k)^2$ então faça $y_k = x_k + \alpha_k p_k^i,\ \widetilde{\alpha}_{k+1} = \alpha_k$ e vá para o passo 6. ;
5		Faça $\widetilde{\alpha}_{k+1} = \theta\widetilde{\alpha}_k$ e $y_k = x_k$
6		Encontre x_{k+1} tal que $f(x_{k+1}) \leq f(y_k)$, fazendo $k = k + 1$, e vá para o passo 3

Figura 5.1: Pseudo código para algoritmos genéricos de busca direta

O algoritmo 5.1 mostra que em cada iteração é possível obter algum ponto para o qual existe um decrescimento suficiente da função objetivo identificado no passo 1. O tamanho do passo α_k é reduzido somente quando não é possível identificar uma

redução suficiente de f ao longo da direção de busca p_k^i, para $i = 1, ..., r$ (passos 2-3). No passo 6 o algoritmo aceita algum ponto que produz uma redução da função objetivo em relação ao ponto selecionado y_k.

Deve-se notar no passo 4, alguma técnica de extrapolação que deve ser procurada, para determinar um bom tamanho do passo α_k, sempre que uma direção apropriada de busca tenha sido detectada. Entretanto, o uso de uma técnica de extrapolação não é necessária para garantir a convergência global, bastando fazer $\alpha_k = \widetilde{\alpha}_k$. Além disso, tomando-se um conjunto com r direções de busca $p_i^k,\ i = 1, ..., r$, associada a um ponto corrente x_k, dependendo de quão grande seja o decrescimento ao longo de uma direção p_i^k, as outras direções correntes podem ser ignoradas [37].

Finalmente, observa-se que os passo 1 e 6 admitem a possibilidade do uso de algum esquema de aproximação para função objetivo produzir um novo melhor ponto.

As propriedades do algoritmo são descritas no seguinte teorema:

Teorema 5.2.2 *Conforme [35], faça $\{x_k\}$ ser uma seqüência produzida pelo algoritmo 5.1. Supondo que a seqüência de direções $\{p_k^i\}_{i=1}^r$ satisfazem a Equação (5.2.2), então tem-se que:*

$$\lim_{k\to\infty} \inf \|\nabla f(x_k)\| = 0 \tag{5.2.4}$$

Prova: Para provar o teorema (5.2.2) basta mostrar que as condições (1)–(3) fornecidas pelo teorema (2.7.1) são satisfeitas por uma subseqüência de $\{x_k\}$.

A condição 1 é atendida bastando seguir as instruções do algoritmo. A condição (2) é obviamente verdadeira, restando apenas a condição (3) para provar que:

Existe pelo menos uma seqüência de pontos $\{y_k^i\}$ e de escalares positivos $\{a_k^i\},\ i =$

$1, ..., r$ tal que:

$$f(y_k^i + a_k^i p_k^i) \geq f(y_k^i) - o(a_k^i) \tag{5.2.5}$$

$$\lim_{k\to\infty} a_k^i = 0 \tag{5.2.6}$$

$$\lim_{k\to\infty} \|x_k - y_k^i\| = 0 \tag{5.2.7}$$

$$\text{Então} \quad \lim_{k\to\infty} \|(x_k)\| = 0 \tag{5.2.8}$$

Divide-se a seqüência de iteração $\{k\}$ em três partes K_1, K_2 e K_3, procedendo assim nestas iterações onde os testes nos passos 1, 2, 3, 4 e 5 sejam satisfeitos. Em particular, se $k \in K_1$, tem-se que:

$$f(x_{k+1}) \leq f(x_k) - \gamma\widetilde{\alpha}_k \tag{5.2.9}$$

se $k \in K_2$, tem-se:

$$f(x_{k+1}) \leq f(x_k) - \gamma(\alpha_k)^2 \leq f(x_k) - \gamma(\widetilde{\alpha}_k)^2; \tag{5.2.10}$$

e se $k \in K_3$, tem-se:

$$f(x_k + \widetilde{\alpha}_k p_k^i) > f(x_k) - \gamma(\widetilde{\alpha}_k)^2 \quad \text{para } i = 1, 2.., r \tag{5.2.11}$$

Se K_1 é um subconjunto infinito, então na equação (5.2.5) a condição 1 e hipótese de continuidade da f implicam que:

$$\lim_{k\to\infty,\ k\in K_1} \widetilde{\alpha}_k = 0 \tag{5.2.12}$$

Assumindo que K_2 é um subconjunto infinito, a partir da equação (5.2.6), pelas mesmas razões acima, obtem-se:

$$\lim_{k\to\infty,\ k\in K_2} \widetilde{\alpha}_k = 0 \tag{5.2.13}$$

Agora para cada $k \in K_3$ faça m_k ser o maior índice tal que $m_k < k$ e $m_k \in K_1 \cup K_2$. Então tem-se que:

$$\widetilde{\alpha}_{k+1} = \theta^{k-m_k}\widetilde{\alpha}_{m_k} \tag{5.2.14}$$

Assume-se que $m_k = 0$ se o índice m_k não existir, desde que K_1 e K_2 sejam vazios.

Como $k \to \infty$ e $k \in K_3$, também $m_k \to \infty$ (se $K_1 \cup K_2$ é um subconjunto infinito) ou $(k - m_k) \to \infty$ (se $K_1 \cup K_2$ é finito). Entretanto, a equação (5.2.14), juntamente com a equação (5.2.12) e (5.2.13) ou o fato de $\theta \in (0, 1)$, implica que:

$$\lim_{k\to\infty,\ k\in K_3} \widetilde{\alpha}_k = 0 \tag{5.2.15}$$

Então, usando as equações (5.2.12), (5.2.13) e (5.2.15), pode-se escrever:

$$\lim_{k\to\infty} \widetilde{\alpha}_k = 0 \tag{5.2.16}$$

Da equação (5.2.16) conclui-se que existe um número finito de subconjuntos $K \subseteq \{0,\ 1, ...\}$ tal que $\widetilde{\alpha}_{k+1} < \widetilde{\alpha}_k$ para todo $k \in K$; isto é, o passo 5 é realizado para todo $k \in K$. Conseqüentemente, tem-se que $K_3 \subseteq K$, e então a equação (5.2.11) é aderente para todo $k \in K$. Agora, em relação à condição (3) da Teorema (2.6.1), faça para cada $k \in K$ o conjunto:

$$a_k^i = \widetilde{\alpha}_k, \quad y_k^i = x_k, \quad ß = 1,\ 2, ..., r \tag{5.2.17}$$

$$\text{Então } f(y_k^i + a_k^i + a_k^i p_k^i) \geq f(y_k^i) - \gamma(a_k^i)^2; \tag{5.2.18}$$

entretanto, tomando a equação (5.2.16), obtem-se que:

$$\lim_{k\to\infty,\ k\in K} a_k^i = 0 \tag{5.2.19}$$

e portanto as equações (5.2.5) e (5.2.6) aderem. Finalmente, a equação (5.2.7) é conseqüência direta de (5.2.17), concluindo assim a prova.

Agora definindo-se um algoritmo de busca que produz seqüências de pontos com a propriedade de que cada ponto limite é um ponto estacionário de f, esta propriedade adicional pode ser obtida pela investigação de mais detalhes sobre o comportamento da função objetivo ao longo da direção de busca $p_k^i,\ i = 1, ..., r$, e pelo uso de técnica de busca de direção sem derivada para assegurar movimentos suficientemente longos ao longo de alguma direção de uma boa direção identificada pelo algoritmo, conforme figura (5.2).

PASSOS		Procedimentos
1	Dados	$x_0 \in \mathbb{R}^n,\ \widetilde{\alpha}_0 > 0,\ i = 1, ..., r,\ \gamma > 0,\ \delta, \theta \in (0, 1)$
2		Faça $k = 0$
3		Faça $i = 1$ e $y_k^i = x_k$
4		Se $f(y_k^i + \widetilde{\alpha}_k^i p_k^i) \leq f(y_k^i) - \gamma(\widetilde{\alpha}_k^i)^2$, então calcule $\alpha_k^i = \min\{\delta^{-j}\widetilde{\alpha}_k : j = 0, 1, ...\}$ tal que $f(y_k^i + \alpha_k^i p_k^i) \leq f(x_k) - \gamma(\alpha_k^i)^2$ e faça $\widetilde{\alpha}_{k+1}^i = \alpha_k^i$; caso contrário faça $\alpha_k^i = 0$ e $\widetilde{\alpha}_{k+1}^i = \theta\widetilde{\alpha}_k^i$. Faça $y_k^{i+1} = y_k^i + \alpha_k^i p_k^i$
5		Faça $\widetilde{\alpha}_{k+1} = \theta\widetilde{\alpha}_k$ e $y_k = x_k$
6		Se $i < r$, faça $i = i + 1$ e vá para o passo 4
7		Encontre x_{k+1} tal que $f(x_{k+1}) \leq f(y_k^{r+1})$, faça $k = k + 1$, e vá para o passo 3

Figura 5.2: - Algoritmo de busca direta com movimentos controlados na direção do ótimo

Em cada iteração k o algoritmo examina o comportamento da função objetivo ao longo da direção da busca $p_k^i,\ i = 1, ..., r$ (Passos 3-5). Entretanto, sempre que o algoritmo detecta a direção p_k^i onde a função é suficientemente decrescente, o algoritmo produz um novo ponto, avaliando o desempenho da função, ajustando-se um valor suficientemente grande para mover-se ao longo desta direção. Este ponto é determinado pela média dos tamanhos de passos apropriados α_k^i, veja passo 4. No passo 7, similarmente ao algoritmo genérico Figura (5.1) , o novo ponto x_{k+1} pode ser o ponto y_k^{r+1} produzido nos passos de 3-5, ou é algum ponto em que a função objetivo é melhorada em relação a $f(y_k^{r+1})$. Este fato, conduz a adoção de um

esquema de aproximação para a função objetivo produzir um novo e melhor ponto e então aperfeiçoar a eficiência do algoritmo sem afetar as propriedades de convergência.

Comparando o algoritmo 1 (figura 5.1) com o 2 (figura 5.2) é fácil perceber que o algoritmo 2 requer condições fortes para produzir o novo ponto. De fato, todas as direções devem ser investigadas em cada iteração. Entretanto, no algoritmo 2 é possível associar à cada direção p_k^i um tamanho de passo inicial $\widetilde{\alpha}_k^i$, na qual é obtido com base no comportamento da função objetivo ao longo do p_k^i observado na iteração vigente.

Neste caso, as instruções do algoritmo devem garantir que os passos iniciais $\widetilde{\alpha}_k^i,\ i = 1, 2, ..., r$, levem em conta os diferentes comportamentos de f ao longo da direção de busca.

Finalmente, no algoritmo 2, nota-se a necessidade de examinar em cada iteração o comportamento de f ao longo de todas as r direções. Entretanto, para cada iteração o ponto corrente x_k é obtido pela média de pontos intermediários y_k^{i+1} sempre que um decrescimento suficiente de f seja obtido ao longo de alguma das direções de busca $p_k^i, i \in \{1, ..., r\}$.

Através do algoritmo 1 e 2 pode-se estabelecer uma teoria geral de convergência para otimização sem restrição e sem a utilização de derivadas. Com este objetivo, pode-se estabelecer um conjunto de condições tais que satisfaçam um padrão de busca garantindo a convergência global. Sobre a base de análise teórica, define-se novos padrões de algoritmos de busca direta nos quais combinam aspectos das hipóteses da busca em linha com padrão de busca.

Como foi verificado na seção (5.2.1) que a direção da busca e tamanho do passo além do critério de convergência são os fatores diferenciadores dos aspectos construti-

vos dos diversos algoritmos. Desta forma, a partir de agora é apresentada a descrição do TALUS ressaltando os seus aspectos diferenciais dos demais algoritmos de busca direta.

5.3 Construindo o TALUS

No processo de construção discute-se o problema comum dos algoritmos de busca direta para OG [19]. A partir das dificuldades identificadas em relação à eficiência dos algoritmos existentes, um novo algoritmo probabilístico foi construído para resolver problemas de OG, superando as dificuldades identificadas.

Em geral, os modelos de busca direta fazem parte da classe dos algoritmos que não utilizam derivadas. Embora possuam condições de convergência, a escolha da direção ótima de busca apresenta dificuldades de ser obtida. Devido a estas dificuldades existe um grande número de algoritmos de busca direta na tentativa de melhorar a eficiência da busca da direção do ótimo. Uma outra característica desejada nos algoritmos de otimização é a sua abrangência. É comum obter-se algoritmos com uma eficiência aceitável para uma determinada classe de problemas de OG e de baixa eficiência para as demais classes, por isso busca-se algoritmos abrangentes e que sejam de fácil implementação e uso.

5.3.1 Fundamentos Funcionais do TALUS

Otimização Global tem sido um campo de pesquisa bastante ativo. Não há dúvidas de que isto é motivado pelo fato de que os problemas de OG surgem em grande variedade de áreas e contextos. Portanto, é evidente a busca de algoritmos para

resovê-los [10], [35], [1], [98], [47].

O problema geral pode ser posto como descrito a seguir:

$$\text{Max } f(x) \tag{5.3.1}$$

$$\text{sujeito à: } x \in S$$

onde f é uma função contínua e convexa sobre S, e $S \subset \mathbb{R}^n$ é um conjunto compacto. Em outras palavras, foca-se o problema em um problema de OG irrestrito.

Como visto, muitos algoritmos determinísticos são mais recomendados para algumas classes de problemas, apresentando uma eficiência muito boa quando trata-se de problemas onde o espaço de busca pode ser tratado como convexo e a função objetivo é contínua e diferenciável, com muitas hipóteses tendo sido propostas [10], [49], [99], [100] e [101]. Considerando que a condição de otimalidade obriga a função objetivo ser contínua e diferenciável e o espaço de busca convexo, no mundo real esta condição é restritiva, porque, em muitos casos reais, a função objetivo não é diferenciável ou não pode ser calculada ou explicitamente aproximada; e ainda mais, o espaço de busca não é convexo.

As características do TALUS são as seguintes:

- O algoritmo trabalha e converge mesmo que f não seja contínua. Se f for contínua, é fácil encontrar o critério de fim de convergência.

  O número de pontos de ótimos globais pode ser ilimitado, porém f tem de ser limitada em uma região na qual será otimizada.

- O algoritmo é de fácil implementação, os passos básicos são responsáveis por obter soluções para o problema, ou para obter uma solução aproximada quando os cálculos são feitos através do computador.

As hipóteses de aceleração são implementadas considerando que a direção de busca é obtida a partir de informações sobre o comportamento da função objetivo.

Existe uma grande quantidade de algoritmos baseada em heurísticas na escolha da direção da busca como descrito no Capítulo 4, exemplo o "*Simulated Annealing*". Os da classe evolutiva, os Genéticos, são frequentemente mencionados em muitos contextos como: redes neurais e inteligência artificial [102], [103], entretanto, estes algoritmos utilizam-se da capacidade de processamento do computador tendo como conseqüência uma convergência lenta, quando ocorre. Como apresentando na seção (4.2) a programação de resfriamento do "*simulated annealing*" deve ser lenta para garantir quase sempre a convergência.

As técnicas estão sempre voltadas para identificar o melhor caminho da busca. No proposta do algoritmo TALUS, a nuvem segue inicialmente uma distribuição de probabilidade uniforme no intervalo de busca. Este procedimento permite fazer uma estimativa sobre a topologia da função objetivo. Independentemente dos aspectos da forma da função objetivo são coletadas amostras f utilizando-se da nuvem. Em seguida é selecionado o ponto da nuvem x_i^k que apresenta melhor desempenho, ou seja, de maior valor (no caso de maximização). A partir destas amostras é possível estabelecer uma função de estanciamento sobre a função objetivo de forma que esta função possa fornecer uma distribuição de probabilidade da aproximação, dos diversos elementos da nuvem, em relação a um máximo local.

Esta distribuição de probabilidade muda a cada iteração, ela fornece informações sobre o comportamento da nuvem em relação ao ótimo. Desta forma, informa também o posicionamento dos elementos da nuvem em relação a sua média, fazendo com que estes pontos através dessa informação e de um adequado tamanho de passo possa

caminhar na direção do ótimo.

Por outro lado, para determinar o tamanho do passo deve-se considerar alguns aspectos:

- Os passos iniciais devem ser suficientemente pequenos para que se possa avaliar a vizinhança dos pontos iniciais, gerados pela distribuição uniforme. É claro que, se o tamanho da nuvem for suficientemente grande, permitirá uma rápida estimativa sobre a forma da função objetivo.

- A direção do passo é obtido pela assimetria que garante que os valores de $f(x_i^k) < f(\overline{x}^k - a^k)$ receberão valores positivos e caso contrário valores negativos, indicando assim a direção do passo. Para determinar o tamanho do passo, ganho e direção, é utilizada uma função de estanciamento (monotonicamente crescente) da função objetivo. Esta função permitirá estabelecer uma distribuição de probabilidade para cada $f(x_i^k)$ sendo assim possível estabelecer os momentos desta distribuição.

- O critério de convergência é baseado no fato de que: o conjunto $F_{\max}^k$ é uma sucessão crescente de pontos que fazendo k tender para infinito obtém-se o máximo absoluto de f se houver, pois, $\lim_{k\to\infty} \operatorname{Max} f(x_i^k) = \operatorname{Max} F\{f(x_i^k)\} \approx \operatorname{Max} f(x)$.

A assimetria e média são consideradas por este processo como os mais importantes. As diferentes hipóteses associadas a este procedimento garantem que, usando informações sobre a forma da função objetivo em diversas iterações, são capazes de mapear o formato de f e conseqüentemente os seus pontos de descontinuidade, estacionários e pontos de ótimo, quando houver.

Somente a primeira nuvem é gerada de acordo a distribuição uniforme sobre o conjunto de busca S. O tamanho da nuvem é um valor empírico, no qual, em geral o seu valor eqüivale a 100 vezes o número de variáveis, desta forma obtém-se uma boa *performance*. Conforme [2] cada ponto desta nuvem move-se por parâmetros da distribuição de probabilidade produzido pela função $F(f(x_i^k))$ estanciamento, conforme Figura (5.3). O valor esperado de sucessivas distribuições de probabilidade constroi um caminho que irá convergir para o ótimo global.

5.3.2 O Caso Unidimensional

A idéia básica do algoritmo é trabalhar a função objetivo como uma função densidade de probabilidade sobre os pontos do conjunto de busca. Um dos cuidados encontra-se na observação da moda desta distribuição. O cuidado sobre a moda deve ser tomado pelo fato de f poder assumir valores negativos, e também concentrar a massa probabilística ao redor da moda, uma função de transformação apropriada de f deve ser usada para criar uma distribuição de probabilidade. O procedimento para obter a função de transformação é descrito a seguir.

O primeiro passo é criar uma nuvem de r números aleatórios $x_i^k,\ i = 1, 2..., r$, uniformemente distribuida sobre o intervalo de busca onde tem como pontos limites do intervalo a e b. Em todas as iterações do algoritmo, os pontos da nuvem irão mover-se com um passo a ser definido. Então se define uma distribuição de probabilidade discreta para os pontos da nuvem para k^{th} iterações do algoritmo da seguinte forma:

$$F^k(x_i^k) = \frac{1}{1 + k^{2\delta \frac{f_{\max}^k - f^k(x_i^k)}{f_{\max}^k + 10^{-10}}}} \tag{5.3.2}$$

onde $\delta(k) > 0$ e o seu valor típico é $\delta = 1$.

Pode-se fazer com que $\delta(k)$ tenha um crescimento em função de k fazendo com que a função F^k tenha uma evolução diferente.

Para valores de x_i para os quais $f^k(x_i^k)$ não seja definida assume-se que $F_i^k = 0$ desta forma independentemente da função objetivo, ser contínua ou não, sempre haverá uma $F_i^k(x_i^k)$.

Os de valores de $f_{\max}$ são dados por:

$$f_{\max}^k = \max f(x_i^k) = \max\{f(x_1^k), f(x_2^k), ...f(x_r^k)\}$$

Como F^k nem sempre assume valores positivos, define-se que a distribuição de probabilidade para os pontos da nuvem em k^{th} iterações como:

$$p_i^k = \frac{|F^k(x_i^k)|}{|\sum_{j=1}^{r} F^k(x_j^k)|} \tag{5.3.3}$$

onde $i, j = 1, 2....r$.

A partir da expressão F^k pode-se observar que o maior valor de p_i^k coresponde a x_i^k que fornece o maior valor de f, que é $(f_{\max}^k)$, fornecido por um x_i^k que chama-se de $x_{\max}^k$. Para um k apropriadamente grande a distribuição $\{p_i^k\}$, $i = 1, 2..., r$ obtém-se uma alta concentração ao redor de $x_{\max}^k$. O valor esperado da distribuição $\{p_i^k\}$, dada por:

$$\overline{x}_F^k = \sum_{i=1}^{r} x_i^k p_i^k \tag{5.3.4}$$

O valor esperado dado acima fornece informações sobre o comporatamento da função distribuição de probabilidade de uma determinada iteração. Considerando que a distribuição das médias das diversas amostras é uma distribuição Normal, desta forma, podemos estimar os parâmetros dessa distribuição como sendo representativa

da função objetivo. Logo o valor do ponto de ótimo terá como estimativa do ponto global x^*.

Os valores das estimativas do desvio padrão e da assimetria da distribuição fornecem, respectivamente:

$$\sigma^k = \sqrt{\sum_{i=1}^{r}(x_i^k - \overline{x}_F^k)^2 p_i^k} \tag{5.3.5}$$

e

$$a^k = \sqrt[3]{\sum_{i=1}^{r}(x_i^k - \overline{x}_F^k)^3 p_i^k} \tag{5.3.6}$$

A assimetria a^k será um guia para indicar em qual lado (em relação a média) a estimativa dos valor da moda se encontram.

O passo para cada ponto da nuvem é definido por:

Ganho de passo = GS= $\gamma(k) = \frac{k \cdot \gamma_1}{k+\gamma_2}$ onde o valor típico de $\gamma_1 = 1$ e $\gamma_2 = 100$, variando de acordo com a precisão desejada e número de iterações a serem realizadas, ou seja γ_2, está associado à precisão desejada e associado ao compromisso da quantidade de iterações a serem realizadas, que deve ser pelo menos 100 vezes o valor de γ_2.

O critério para garantir a estabilidade numérica quando da aproximação do ótimo esta associado ao Curare, que tem por objetivo fazer com que não haja evolução da f_i^{k+1} quando atingindo o ponto de ótimo, o Curare = C é dado pela seguinte equação:

$$C_i = 0 \text{ se } f(x_i) = f_{\max}, \text{ e 1 caso contrário} \tag{5.3.7}$$

Sendo combinado com o sinal do passo (direção), o ganho do passo (γ) e tamanho do passo (distância entre o valor médio $\overline{x}^k$ menos a assimetria no valor x_i^k).

O sinal do passo ou direção é obtida apartir da seguinte expressão:

$$\text{Faça } S_i = 1 \text{ se } f(\overline{x} - a) > f(x_i), \text{ e } -1 \text{ caso contrário} \tag{5.3.8}$$

O tamanho do passo depende da seguinte expressão: $(\overline{x} - a - x_i)$.

Logo, o passo a ser dado varia para cada ponto da nuvem, fazendo um processo de caminhada mais lenta no ínicio e mais rápido ao final. Esta caminhada mais lenta, permite a percepção da existência de uma direção mais apropriada para obtenção do ótimo global.

A expressão que representa o tamanho do passo é dada por:

$$q_i = \gamma_k \cdot C_i \cdot S_i \cdot (\overline{x} - a - x_i) \tag{5.3.9}$$

Desta forma define-se a nova nuvem a partir da nuvem anterior levando-se em conta o comportamento dos parâmetros fornecidos pelas amostras da função objetivo.

$$x_i^{k+1} = x_i^k + \gamma_k \cdot C_i \cdot S_i \cdot (\overline{x} - a - x_i) \tag{5.3.10}$$

Caso x_i^{k+1} fique fora do intervalo de busca se faz a sua substituição por $\overline{x}$.

Observa-se que $\gamma(k)$ possui um valor muito pequeno para valores pequenos de k e cresce monotonicamente para valores de k quando tendem para infinito. O passo q é construído de tal forma que se $f^k(x_i^k) = f_{\max}^k$ então $q_i^k = 0$.

Se $x_i^k = x_{\max}^k$ então $x_i^{k+1} = x_{\max}^k$, isto é, x_i^k não se move. Se $(\overline{x}_F^k - a^k) - x_i^k > 0$ e $f(\overline{x}_F^k - a^k) - f(x_i^k) > 0$ à medida em que x_i^k estiver localizado do lado esquerdo da região onde o máximo global se encontra, então esses pontos serão movidos para a direita, conforme equações (5.3.7–5.3.10). Agora se $(\overline{x}_F^k - a^k) - x_i^k < 0$ e $f(\overline{x}_F^k - a^k) - f(x_i^k) > 0$ neste caso, x_i^k está localizado do lado direito da região onde o máximo global se encontra, então esses pontos serão movidos para a esquerda, conforme equações (5.3.7–5.3.10). Os outros dois casos são análogos.

A função $\gamma(k)$ é tal que nos primeiros passos os ganhos são pequenos, de modo que não se perca a possibilidade de se identificar uma estreita região onde possa conter o máximo. Uma forma de controlar isto é ajustando γ em função do tamanho da nuvem e da quantidade de passos, assumindo-se uma probabilidade de se perder um máximo global localizado em uma região de largura específica do conjunto de busca.

Se $f_{\max}^{k+1} > f_{max}^{k}$ a distribuição $\{p_i^{k+1}\}$ será concentrada sobre $x_{\max}^{k+1}$, a qual pode ser completamente diferente da originada por $x_{\max}^{k}$. O algoritmo se comporta para $f_{\max}^{k} = f_{\max}^{k+1}$, o ponto $x_{\max}^{k}$ não irá se mover, como já foi visto $x_{\max}^{k} = x_{\max}^{k+1}$, neste caso o Curare é igual a zero.

Os parâmetros internos δ, γ, γ_1 e γ_2 são para ajuste da melhoria da precisão do algoritmo, objeto de discussão no Capítulo 6.

À medida que k cresce, a nuvem de pontos randômicos concentra-se mais e mais ao redor do máximo global. Se em algum caso, o novo ponto da nuvem extrapolar os limites dos intervalos de busca , deve ser considerado este ponto como tendo violado o intervalo e portanto o ponto substituido pelo valor médio $\overline{x}^k$ da iteração atual, pois, o valor representado por $\overline{x}^k$ é o melhor adaptado.

Para grandes valores de n e pequenos tamanhos de passos torna-se grande a probabilidade da seqüência $\{\overline{x}_F^k\}$ convergir para x^*. Estas duas condições são facilmente controladas pelos parâmetros γ_1, γ_2 e o tamanho da nuvem. Estes aspectos são discutidos no Capítulo 6.

5.3.3 O Caso n-Dimensional

Neste caso o algoritmo trabalha como se estivesse otimizando $n-$unidimensionais funções, pesquisando o máximo em todas as dimensões. De fato, em cada passo, pode-

se pensar n funções delimitadas pelas suas projeções em nuvens de n-dimensões (de tamanho m) sobre n-dimensões. Para adaptar o algoritmo a n-dimensões implementa-se a seguinte expressão:

$$F^k(x_{1i}^k, x_{2i}^k, ..., x_{ni}^k) = \frac{1}{1 + k^{2\delta} \frac{f_{\max}^k(x_{1i}^k, x_{2i}^k, ..., x_{ni}^k) - f^k(x_{1i}^k, x_{2i}^k, ..., x_{ni}^k)}{f_{\max}^k(x_{1i}^k, x_{2i}^k, ..., x_{ni}^k) + 10^{-10}}} \tag{5.3.11}$$

onde $\delta(k) > 0$ e o seu valor típico é $\delta = 1$.

Transformando as demais equações para o caso n-dimensional temos:

$$\prod(x_{1i}^k) = p^k(x_{1i}, x_{2i}, ..., x_{ni}) = \frac{|F(x_{1i}, x_{2i}, ...x_{ni})|}{|\sum_{i=1}^{m} F^k(x_{1i}, x_{2i}, ...x_{ni})|} \tag{5.3.12}$$

$$\overline{x}_{1F}^k = \sum_{i=1}^{m} x_{1i}^k \prod(x_{1i}^k) = \sum_{i=1}^{m} x_{1i}^k \frac{|F^k(x_{1i}^k, ..., x_{ni}^k)|}{|\sum_{j=1}^{m} F^k(x_{1j}^k, ..., x_{nj}^k)|} \tag{5.3.13}$$

$$\overline{x}_{nF}^k = \sum_{i=1}^{m} x_{ni}^k \prod(x_{ni}^k) = \sum_{i=1}^{m} x_{ni}^k \frac{|F^k(x_{1i}^k, ..., x_{ni}^k)|}{|\sum_{j=1}^{m} F^k(x_{1j}^k, ..., x_{nj}^k)|} \tag{5.3.14}$$

$$\text{Ganho do passo} = \gamma(k) \tag{5.3.15}$$

$$C_i = 0 \text{ se } f^k(x_{1i}, x_{2i}, ..., x_{ni}) = f_{\max}^k \text{ e 1 caso contrário} \tag{5.3.16}$$

Sendo combinado com o sinal do passo (direção), ganho do passo (γ) e tamanho do passo (distância entre o valor médio $\overline{x}^k$ menos a assimetria e valor x_i^k.

Abaixo estão as equações relacionadas com o sinal

$$a_1^k = \sqrt[3]{\sum_{i=1}^{m} (x_{li}^k - \overline{x}_{1F}^k)^3 \Pi_i^k} \tag{5.3.17}$$

$$a_n^k = \sqrt[3]{\sum_{i=1}^{m} (x_{ni}^k - \overline{x}_{nF}^k)^3 \Pi_i^k} \tag{5.3.18}$$

$$\text{Sinal ou direção 1} = \frac{f(x_{1F}^k - a_1^k, ..., x_{ni}^k) - f(x_{1i}^k, ..., x_{ni}^k)}{|f(x_{1F}^k - a_1^k, ..., x_{ni}^k) - f(x_{1i}^k, ..., x_{ni}^k)|} \tag{5.3.19}$$

$$\text{Sinal n} = \frac{f(x_{1i}^k, ..., x_{nF}^k - a_n^k) - f(x_{1i}^k, ..., x_{ni}^k)}{|f(x_{1i}^k, ..., x_{nF}^k - a_n^k) - f(x_{1i}^k, ..., x_{ni}^k)|} \tag{5.3.20}$$

$$\text{Tamanho do passo 1} = \gamma_1 \cdot C_1 \cdot S_i \cdot (\overline{x}_{1F}^k - a_1^k) - x_{1i}^k \tag{5.3.21}$$

$$\text{Tamanho do passo n} = \gamma_n \cdot C_n \cdot S_n \cdot (\overline{x}_{nF}^k - a_n^k) - x_{ni}^k \tag{5.3.22}$$

$$x_{1i}^{k+1} = x_{1i}^k + \text{Tamanho passo 1} \tag{5.3.23}$$

$$x_{ni}^{k+1} = x_{ni}^k + \text{Tamanho do passo n} \tag{5.3.24}$$

O distribuição de probabilidade gerada a partir de $\{F^k(x_{1i}^k, ..., x_{ni}^k)\}$ é usada em todas as direções, e isto garante que em todas as dimensões a distribuição irá se concentrar no maior valor de $f(x_{1i}^k, ..., x_{ni}^k)$.

5.3.4 Convergência

Em relação à convergência pode ser dito, que pelas hipóteses assumidas nos três primeiros passos do algoritmo, considerando-se que a geração da nuvem é um vetor puramente aleatório de tamanho n, escolhendo-se uma direção através da obtenção de $f_{\max}$ e fazendo-se r (que é o número de iterações) tender para infinito, este processo passa a ser de busca pura. Cabendo as hipóteses de convergência dadas pelo teorema

5.2.1 [49], a convergência a seqüência $\{f^k_{\max}\}$ é aleatória e monotonicamente não decrescente, e assumindo que é garantida a existência de pelo menos um máximo, esta seqüência converge em probabilidade para o supremo, conforme [24].

O algoritmo garante que a seqüência monotonicamente não decrescente $f^k_{\max}$ na hipótese de f ser limitada sobre um conjunto de busca, é convergente.

Em outras palavras, fazendo $\overline{x}^k_F - x^k_{\max} \to 0$ com $k \to \infty$, nota-se que $\sigma^k \to 0$ e $a^k \to 0$.

Logo a seqüência $\overline{x}^k_F$ é uma seqüência randômica, construída com o objetivo de minimizar:

$$\lim_{k\to\infty} \|\nabla f(x_k)\| = 0 \;\Leftrightarrow\; \lim_{k\to\infty} \sum_{i=1}^{r} \min\{0, \nabla f(x_k)^T p^i_k\} = 0 \tag{5.3.25}$$

conforme definido no início deste capítulo.

Observa-se também que:

$$\lim_{k\to\infty} f^k(x^k_i) = f(x) \tag{5.3.26}$$

logo $\lim_{k\to\infty} \operatorname{Max} f^k(x^k_i) = \operatorname{Max} f(x^*)$

A prova é fornecida pelo teorema (5.2.2).

PASSOS		Procedimentos
1	Dados	$S \subset \mathbb{R}^n$, e $\liminf x_i$, $\limsup x_i \in [a,\ b]$ onde $a, b \in S$ e $i = 1, ..., r$, $\delta =$ constante $= 1$, $\gamma_1 = 1$ e $\gamma_2 = 100$
2		Gere uma nuvem de tamanho r com $x_i \in U \sim [a, b]$ onde $i = 1, ..., r$ e $k = 1$
3		Calcule $$f(x_i^k)$$ e $$\text{Max}\{f(x_i^k)\} = f_{\max}^k$$
4		Calcule $$F\{f(x_i^k)\} = \frac{1}{1 + k^{2\delta}\left(\frac{f_{\max}^k - f_i^k}{f_{\max}^k + 10^{-10}}\right)}$$
5		Calcule $$p_i^k = \frac{\lvert F\{f(x_i^k)\}\rvert}{\lvert \sum_{i=1}^{r} F\{f(x_i^k)\}\rvert}$$
6		Calcule $$\overline{x}_F^k = \sum_{i=1}^{r} x_i^k p_i^k$$ e $$a^k = \sqrt[3]{\sum_{i=1}^{r} (x_i^k - \overline{x}_F^k)^3 p_i^k}$$
7		Faça $C_i^k = 0$ se $f(x_i^k) = f_{\max}^k$ e 1 caso contrário Faça $S_i^k = 1$ se $f(\overline{x}_F^k - a^k) > f(x_i^k)$ e -1 caso contrário
8		Calcule $\gamma^k = \frac{k\gamma_1}{k+\gamma_2}$
9		Faça $q_i^k = \gamma^k \cdot C_i^k \cdot S_i^k \cdot (\overline{x}^k - a^k - x_i^k)$ Se $a \leq x_i^k + q_i \leq b$ faça $x_i^k = q_i + x_k$ e $k = k+1$, caso contrário faça $x_i = \overline{x}^k$
10		Volte para o passo 3

Figura 5.3: - Algoritmo de busca Direta com movimentos controlados - TALUS [2]

Capítulo 6

TESTES E ANÁLISE DE DESEMPENHO DO TALUS

Neste capítulo é realizada uma análise do desempenho do Talus, tomando por base os pressupostos definidos no Capítulo 3. São considerados todos os aspectos que permitem comparar o Talus com algum outro *software* de solução de problemas de OG irrestrito. Busca-se minimizar os problemas relativos à obtenção de uma plataforma, de *hardware* e *software*, que possibilite análises comparativas entre diversos algoritmos. Toma-se como principal meta eliminar a necessidade de adaptações nos algoritmos para receberem entradas de novas funções de teste, ou de alterações de parâmetros internos para avaliação de performance.

Neste sentido o Talus foi desenvolvido em uma versão de *software* para plataforma Windows usando como compilador o DELPHI. A sua estrutura permite analisar qualquer função objetivo usando o interpretador associado ao algoritmo.

Como descrito no Capítulo 3 as etapas da definição do modelo e ambiente de desenvolvimento estão propostas conforme a seguir:

- **Identificação**: formulação dos principais objetivos do modelo e determinação (seleção) da estrutura adequada do modelo. Como foi visto no Capítulo 4 onde são estudadas as hipóteses para construção de modelos, determina-se onde pesquisar, identificando-se as dificuldades existentes. No Capítulo 5 abordou-se diretamente os aspectos construtivos do Talus, discutindo-se funcionalmente cada passo do algoritmo para garantir a cobertura de todos os pressupostos colocados no Capítulo 4.

- **Calibração**: Ajuste do modelo aos dados e resultados conhecidos (engenharia reversa). A calibração está associda ao teste de performance a ser discutido a partir dos resultados obtidos com funções de teste. Em [19] o conjunto de funções de teste é um conjunto reconhecido como *Benchmarking*, aceito pela comunidade de Otimização Global, conforme é discutido nas seções seguintes deste Capítulo.

- **Validação, aplicação e análise**: *forecasting*, controle, e gerenciamento: Todos os testes são aplicados em funções cujos resultados já são conhecidos, e representam problemas, na maioria das vezes, com uma alta complexidade na solução. A análise aqui é feita no ajuste dos parâmetros internos através de procedimentos estatísticos que garantam uma alta precisão e um bom condicionamento numérico.

Como resultado é apresentada a adequação da melhor parametrização do algoritmo permitindo o controle sobre o sistema nas condições mais diversas.

6.1 Implementação do Algoritmo – TALUS

O *software* foi desenvolvido para ser o mais amigável possível, permitindo o fácil entendimento do seu uso e acompanhamento da obtenção do resultado, que também é de fácil interpretação. Possui saídas gráficas com alguns exemplos de aplicações para facilitar o entendimento, além de testes prévios da função objetivo a ser otimizada. Permite a avaliação da sintaxe da função objetivo e do intervalo de busca garantindo uma análise prévia da função a ser otimizada e a aderência *a posteriori* dos resultados ao modelo.

Além dos aspectos funcionais perceptíveis ao usuário, o algoritmo foi implementado de tal forma que torne possível a inclusão de novas facilidades. O desenvolvimento do sistema é baseado em programação voltada a objeto. Toda nova facilidade a ser implementada no TALUS ocorre de forma transparente para o usuário, de modo a manter a performance atual, ou seja, as novas facilidades a serem implementadas não causarão perda de desempenho, nem dificuldades de navegação (interações com o usuário).

Vale ressaltar aqui que o desempenho do algoritmo está associado à estabilidade numérica, precisão e tempo de processamento.

6.1.1 Aspectos Construtivos do *Software* TALUS

Tela Principal: Nesta tela, Figura 6.1, o usuário deverá informar a quantidade de variáveis que contém a função, quais são os limites inferiores e superiores de cada variável, o nome de cada variável, a expressão matemática da função objetivo, o tamanho da nuvem e o número de iterações. Após informar estes campos o *software*

executará todos passos descritos na Figura 5.3. Como se pode observar a Tela prin-

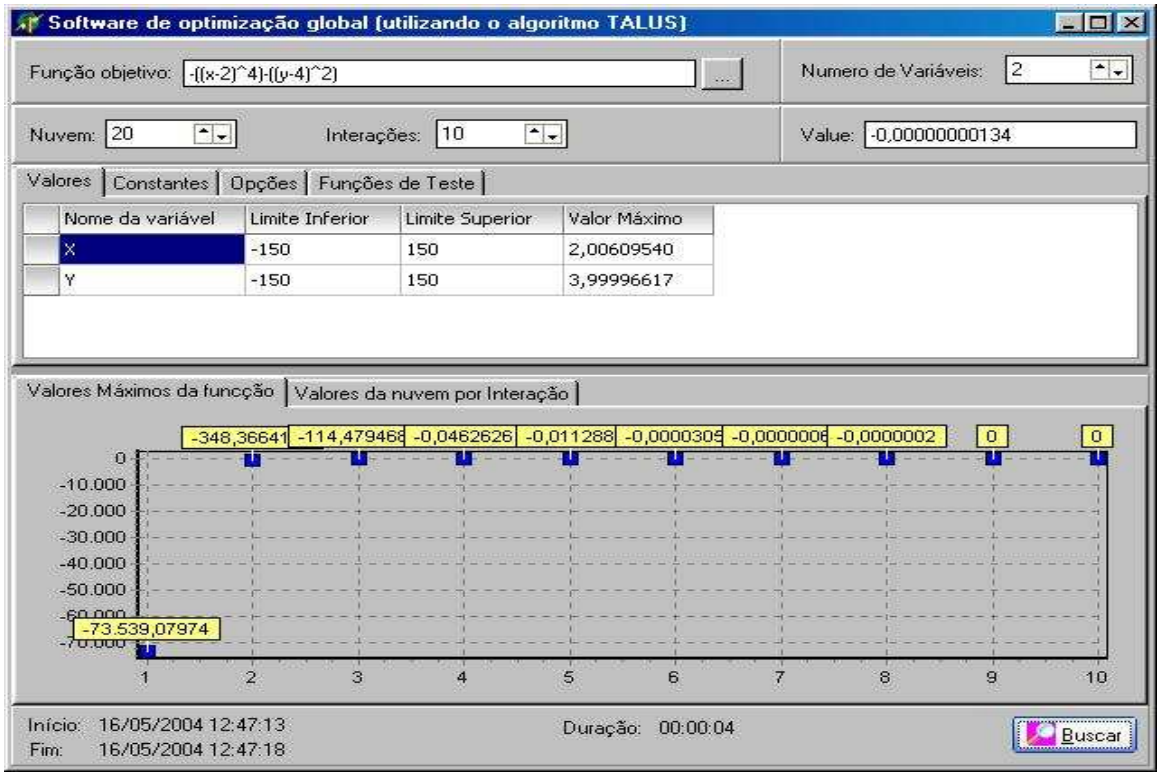

Figura 6.1: - Tela principal do TALUS

cipal do TALUS é de fácil compreensão, pois, são poucos os parâmetros de entrada: identifica-se rapidamente os valores de entrada para o tamanho da nuvem, número de variáveis, intervalos de busca para cada variável, número de iterações desejadas e entrada para função objetivo.

Durante o processamento, na aba "Valores máximos da função" é mostrado um gráfico com os valores encontrados pelo *software* após cada iteração, que é apresentado em tempo real. Esta forma de apresentação é uma ferramenta que possibilita visualizar o comportamento do algoritmo ao longo de cada iteração.

Na tela principal pode-se observar uma aba "Valores da nuvem", onde pode ser observado um gráfico para todos os valores da nuvem após cada iteração. É possível então verificar o momento em que ocorre a convergência para cada variável.

Esta tela permite avaliar dinamicamente como os valores da nuvem vão convergindo. Para cada variável da função é criado um gráfico em tempo de execução. Esta opção faz com que o *software* fique lento, pois, o gráfico é realizado em tempo real, veja Figura 6.2.

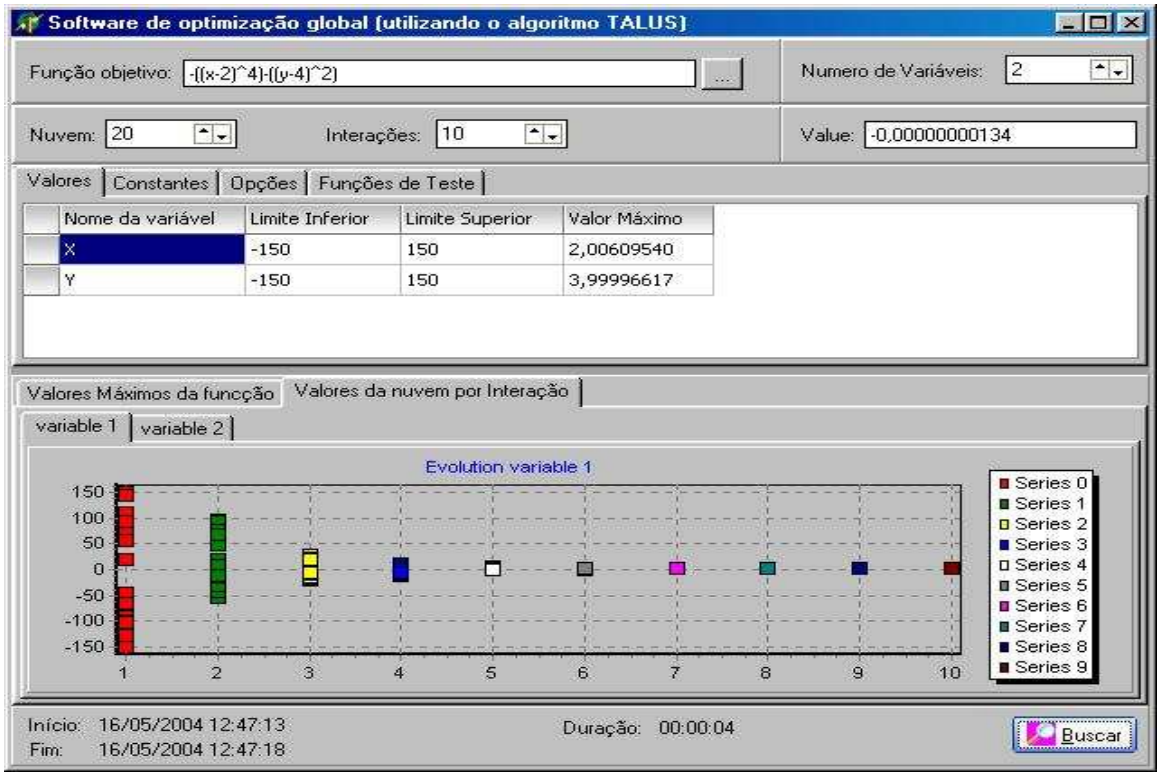

Figura 6.2: Tela convergência das variáveis

Na aba "Opções" o *software* disponibiliza a opção de salvar os valores de cada nuvem em arquivo texto separado por TAB. Esta opção permite fazer análises estatísticas, *a posteriori*, dos valores dos parâmetros internos, assunto tratado na seção 6.2.

Para ser possível a entrada de qualquer função objetivo como dado de entrada é necessário um interpretador de expressões matemáticas. Este interpretador é apresentado como uma tela, chamada pelo botão ao lado da janela "Função Objetivo" na tela principal como na Figura 6.1. Esta tela, Figura 6.3, é uma chamada ao assis-

tente para a construção de expressões matemáticas, que será a função objetivo a ser otimizada pelo algoritmo. Outra característica desta tela é verificar se a função está sintaticamente correta para que o *software* possa interpretá-la. Além destas características o TALUS funciona como um assistente para construção das funções objetivos através do botão de funções, onde se obtém diversas funções predefinidas.

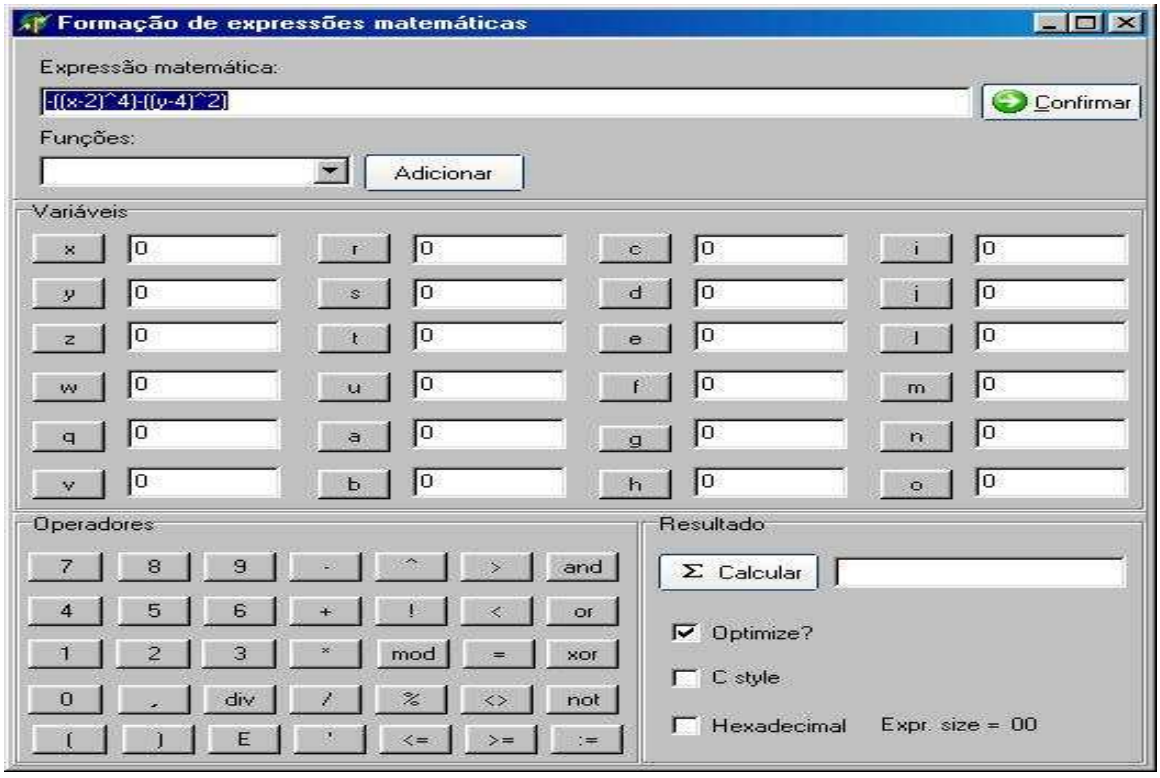

Figura 6.3: - Tela Interpretador de funções

Com relação a sua estrutura lógica construtiva do TALUS é utilizada a programação voltada para objeto. Definem-se variáveis universais e locais de modo a garantir a fácil implementação de novas funcionalidades sem que se tanha que alterar a sua estrutura básica. Conforme [104], no desenvolvimento deste *software* foram adotados todos os critérios para garantir dois atributos: a qualidade e a disponibilidade do *software*. Estes atributos foram obtidos pelo reduzido número de linhas de código em cada subrotina e pela avaliação exaustiva das rotinas, além de se criar pontos de

auditagem vistos nos resultados parciais. Na Figura 6.4, observa-se a estrutura das variáveis e suas relações.

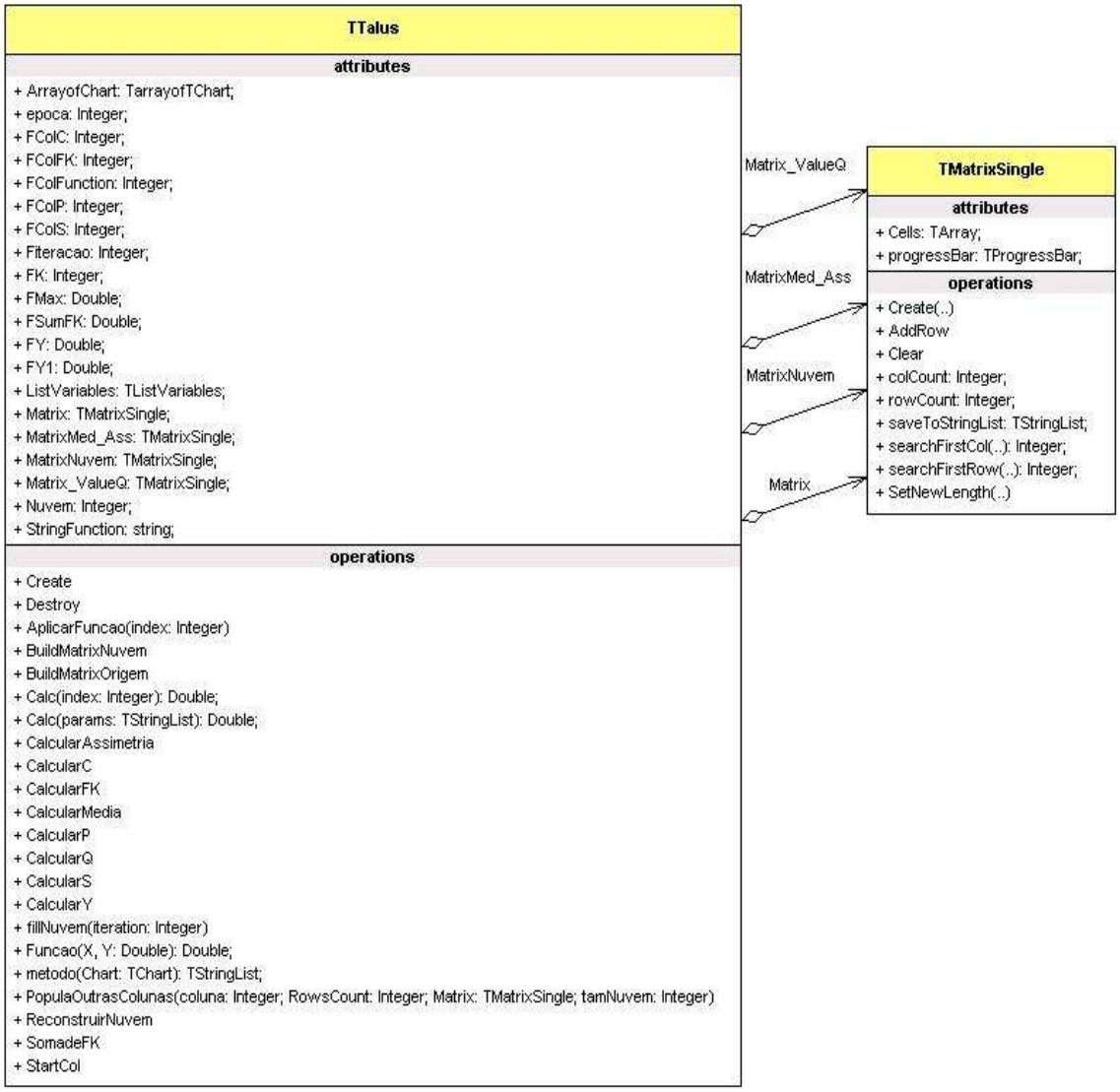

Figura 6.4: - Relacionamento das variáveis

6.1.2 Aspectos Funcionais

O atual estágio de desenvolvimento do TALUS conforme descrito nas pesquisas em [2], [19], [17], [9] vem demonstrando bons resultados.

Os aspectos funcionais internos que garantem uma boa precisão não foram es-

gotados, nem tão pouco as características comparáveis entre os diversos *softwares* existentes. Assim sendo, procura-se destacar na seção 6.2 a avaliação interna do seu desempenho. Na seção 6.3 procura-se fazer testes com funções de referência, esgotando-se então os possíveis questionamentos sobre as características de desempenho e funcionalidades do TALUS.

Uma boa análise da aplicabilidade dos *softwares* de OG é apresentada em [21], onde se observa que a grande maioria não cobre todas as classes de problemas. Embora que como foi visto no Capítulo 3 seção 3.4.1, atualmente existam mais de 3.000 *softwares* para solução de problemas de OG, tomou-se por base a referência [21] por ser a melhor estruturada. Nesta referência observa-se que o *software* LANCELOT cobre praticamente quase todas as classes de problemas de OG. O pacote LANCELOT tem por base o Lagrangeano para lidar com todas as restrições. Suporta solução linear direta e iterativa, uma variedade de condições prévias para quase-Netwon e Métodos de Newton e previsão para diferenças finitas dos gradientes.

Observa-se, no entanto, que o pressuposto básico para a aplicação do pacote está associado as classes de problemas de OG cuja função objetivo seja diferenciável, e que o espaço de busca seja contínuo: pressuposto que restringe significativamente a sua aplicação em situações mais complexas e de natureza mais real.

Quanto ao TALUS é um algoritmo classificado como sendo de busca direta, Para os algoritmos de busca direta estabelecer um critério eficiente de busca da direção do ótimo é o grande diferenciador. Em geral esses métodos estão baseados em heurísticas com parâmetros internos definidos de forma experimental e empírica. Embora esta definição seja válida para a maioria dos algoritmos, no caso do TALUS é reconhecida a estratégia da busca da direção do ótimo por procedimentos estatísticos. Baseando-

se em parâmetros de funções de distribuição de probabilidade e da convergência das seqüências de funções $\{f^n_{\text{Max}}\}$ para $n \leq k$, o algoritmo é monotonicamente crescente e garante que se $k \to \infty$ então f^k_{Max} será o ótimo global, pois f^{k+1}_{Max} só substituirá o f^k_{Max} se for maior, caso contrário f^{k+1}_{Max} permanece f^k_{Max}.

O procedimento para estabelecer uma melhor precisão quanto aos resultados são estabelecidos por seus parâmetros internos, a saber:

$$F\{f(x^k_i)\} = \frac{1}{1 + k^{2\delta}\left(\frac{f^k_{\max} - f^k_i}{f^k_{\max} + 10^{-10}}\right)}$$

onde $k^{2\delta}$ permitirá que a convergência se comporte de forma mais lenta na medida em que se aumentam os valores de δ associandos ao ganho do passo γ pelo número de iterações. A combinação entre estes parâmetros permite uma maior precisão em função da forma da função objetivo. Como em geral a forma da função objetivo não é conhecida, assume-se valores para estes parâmetros conforme regras descritas a seguir.

Nos procedimentos para solução de problemas de OG deve-se levar em conta que o TALUS é um algoritmo probabilístico. Desta forma deve ser realizada uma bateria de resoluções para então se obter o valor da solução. Para aferir os parâmetros internos do TALUS foi utilizada a função de teste de Easom definida em um espaço de busca extenso e que apresenta um único ótimo em uma região muito estreita, Figura 6.5. Estas características impõem dificuldades na busca direta uma vez que nada se sabe sobre a topologia da função a ser otimizada.

Para aferir o desempenho do algoritmo considera-se um intervalo de confiança para os resultados obtidos para os ótimos ao final de um certo número de inteirada. Conforme Figura 6.5, o desvio padrão obtido após 40 interações é de 0,00287 e o valor médio é de 0,998732 com uma nuvem de 100 e o número de iterações igual

a 200. O erro obtido é da ordem do desvio padrão, pois, o valor máximo real da função é igual a 1. É evidente que após um certo número de iterações os valores das variáveis convergem para uma faixa mais estreita do intervalo de busca. Assim sendo se pode melhorar a precisão usando o intervalo de busca não mais para um intervalo tão amplo, e sim para um intervalo bem estreito de tal sorte que o que a pesquisa será realizada em um intervalo cuja probabilidade de se obter o ótimo dezenas de vezes maior que no intervalo inicial, daí ser um procedimento prático para aumentar a precisão do algoritmo.

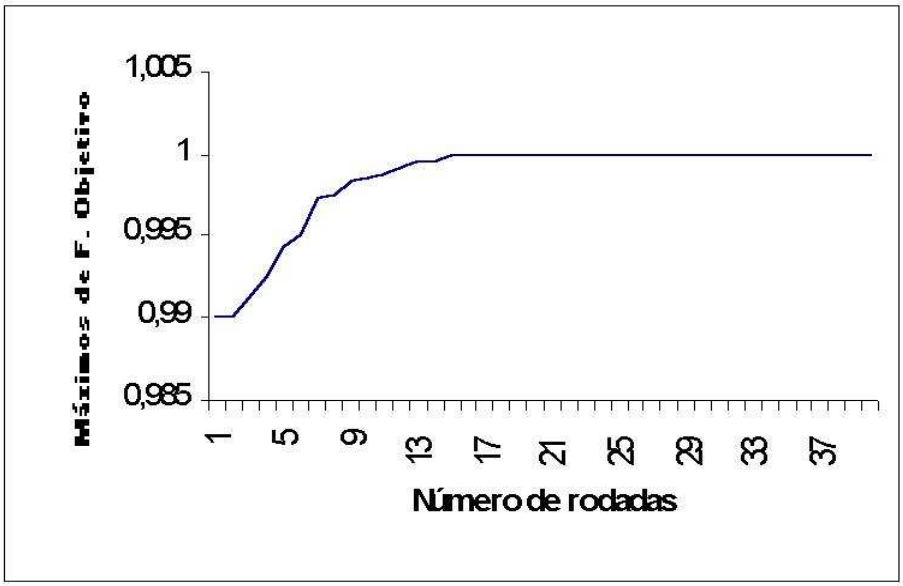

Figura 6.5: - Avaliação do comportamento do TALUS em 40 rodadas com 100 iterações e nuvem igual a 100

Sabe-se que o TALUS é um algoritmo probabilístico e sua convergência para a solução ótima ocorre de forma aleatória. Perceber-se que para o problema em questão a convergência ocorre próximo de 25 iterações, com uma nuvem de 50 pontos e um número de iterações limitado a 100, neste caso não foi definido nenhum critério de

parada a não ser o número de iterações. Como foi definido anteriormente a função de Easom, Figura 6.5, é utilizada para avaliar a efetiva contribuição dos parâmetros internos deste algoritmo, cuja forma é dada por:

$$f(x,y) = -\cos(x)\cdot\cos(y)\exp(-((x-\pi)^2+(y-\pi)^2))$$

sobre o intervalo $-100 \leq x_i \leq 100$, para $i = 1:2$, onde o mínimo global é $f(x,y) = -1$ e $(x^*, y^*) = (\pi, \pi)$

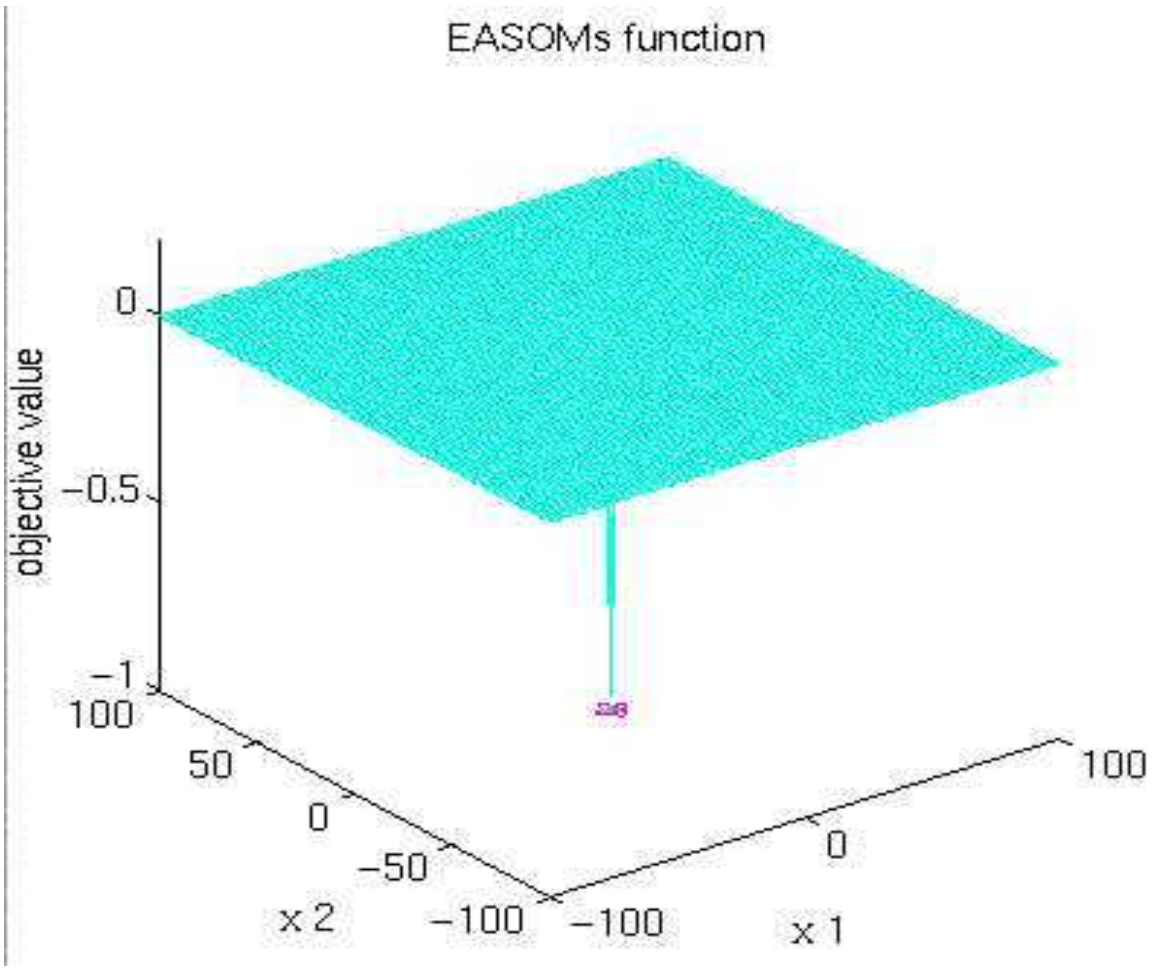

Figura 6.6: - Função usada a avaliação dos parâmetros do TALUS

Aqui deve-se considerar um problema de maximização uma vez que o *software*, TALUS, ainda não foi adaptado para problemas de minimização. Em relação a avaliação do parâmetro δ percebe-se uma grande instabilidade nos valores da nuvem sem haver

garantia de convergência para valores até 100 iterações. A instabilidade da nuvem é provocada pelo ganho de passo com valores iniciais pequenos de modo a permitir uma melhor avaliação na vizinhança dos pontos da nuvem inicial. Essa avaliação consiste em obter maiores informações sobre a topologia da função objetivo. Elevando o valor de δ destaca-se os pontos mais próximos do valor máximo diminuindo a contribuição dos mais distantes no cálculo dos parâmetros da distribuição de probabilidade obtida a partir da função objetivo. Nestes casos quando da existência de muitos ótimos locais é importantes o uso de valores elevados de δ. No caso da função utilizada, a variação neste parâmetro não acrescenta nenhuma informação ao processo.

No ganho do passo $\gamma = \frac{k\gamma_1}{k+\gamma_2}$ garante-se a velocidade em que os valores da nuvem $\{x_i^k\}$ caminham na direção do máximo. Quanto mais estreitos forem os passos iniciais, maior será a quantidade de amostras a se obter da função objetivo para permitir uma melhor avaliação da sua forma, retardando a convergência e explorando melhor a vizinhança de cada ponto formador da nuvem. Na Figura 6.7 mostra o comportamento de γ_2 para os valores de 10, 100 e 1.000, considerando a mesma função de teste com o número de iterações iguais, fixando os demais parâmetros. Nesta Figura pode-se notar que para os valores maiores de γ_2 a convergência é retardada, fazendo com que haja necessidade de se aumentar o número de passos. A relação de compromisso aqui é garantir uma melhor exploração do espaço viável. Fazendo testes exaustivos estima-se o erro esperado por uma relação emprírica dada pela seguinte expressão:

$$\text{Erro} = \frac{\gamma_2^{1/k}}{r} \tag{6.1.1}$$

onde k é o número de iterações e r é o tamanho da nuvem, devendo-se tomar $k > 100 \cdot \gamma_2$ e $k > 100 \cdot$ número de variáveis Utilizando-se os parâmetros conforme acima o erro será minimizado em um procedimento de busca simples. Aplicando-se esta

fórmula obtém-se o erro esperado no procedimento de teste utilizado de 0,0052 contra 0,00287 obtido na prática. Portanto, se pode utilizar a fórmula do erro como sendo o erro previsto pessimista. Além da análise da convergência para o valor de $f_{\max}$

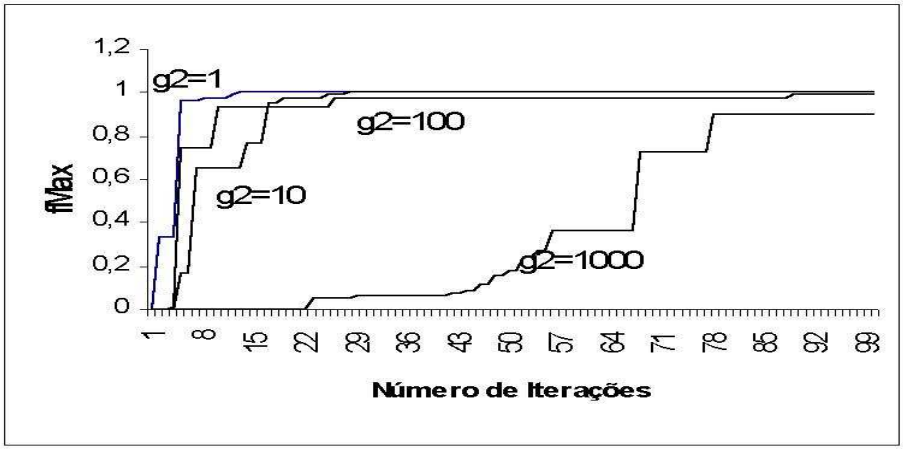

Figura 6.7: - Avaliação do parâmetro gamma2 para valores de 1, 10, 100 e 1000

deve-se observar o comportamento da convergência dos valores das variáveis. Desta forma, se pode garantir que, com os ajustes adequados de parâmetros, a possibilidade de se obter uma melhor performance, por tempo de processamento, por número de iterações ou por precisão. Neste sentido associa-se diretamente o número de iterações com o tempo de processamento.

6.2 Testes de Desempenho do TALUS

Conforme descrito no Capítulo 3, utilizando a função de teste randomizada obteve-se o resultado conforme Tabela 6.1. Pode-se observar que o tempo de processamento cresce com o número de variáveis, pois, como era de se esperar o número de iterações aumenta significativamente até obter a convergência com um erro da ordem de 10^{-9},

conforme descrito em [19] utiliza-se uma nuvem com 200 pontos.

Tabela 6.1: - Avaliação do desempenho do TALUS

SUMÁRIO DOS RESULTADOS DAS FUNÇÕES RANDOMIZADAS		
Número de variáveis	Número de funções de evolução	Tempo de CPU (segundos)
1	6.000	0,1
2	12.000	0,1
3	18.000	0,1
4	24.000	0,1
5	30.000	0,2
6	36.000	0,2
7	42.000	0,2
8	48.000	0,2
9	54.000	0,2
10	60.000	0,3
11	66.000	0,3
12	72.000	0,4
13	78.000	0,4
14	84.000	0,4
15	90.000	0,4
16	96.000	0,6
17	102.000	0,6
18	108.000	0,6
19	116.000	1,0
20	120.000	1,2

$$f(x) = s\sum_{i=1}^{n}(x_i - x_i^*)^2 + \sum_{k=1}^{k_{max}} a_k \,\mathrm{sen}^2[f_k P_k(x - x^*)] \tag{6.2.1}$$

Como a $f(x)$ é definido para um número fixo de n, se segue:

- $s = 0,025n$ fator de escala por definição depende da dimensão do modelo.
- $l = -5,\ u = 5$ limites inferior e superior idênticos para cada componente x.
- x^* solução randomicamente gerada, escolhida por uma distribuição uniforme sobre [l,u].
- $a_k = f_k = 1(k = 1, ..., k_{max})$ escalas de amplitude e múltiplos de freqüência;

$k_{max} = 2$ e portanto:

$$P_1(x - x^*) = \sum_{i=1}^{n}(x_i - x_i^*) + \sum i = 1^n (x_i - x_i^*)^2 \tag{6.2.2}$$

$$P_2(x - x^*) = \sum_{i=1}^{n}(x_i - x_i^*) \tag{6.2.3}$$

são termos de ruído polinomial.

Neste caso a função de teste foi testada para os números de variáveis de 1 até 20. O número de funções de evolução estão associadas ao número de operações com a função objetivo e o tempo de CPU com o tempo necessário para o algoritmo obter o resultado. Observa-se nesta etapa que o tempo de interpretação da função objetivo encontra-se contabilizado neste tempo.

Para consolidar a uso deste algoritmo foram utilizadas também algumas funções do conjunto de Moré considerado *benchmarking*, a seguir mostradas suas formas, Figuras de 6.8 – 6.12:

A função De Jong é uma função simples continua, convexa e unimodal, enquanto a função Ackley's Path é uma função muito utilizada para teste uma vez que é multimodal.

$$\text{De Jong's } f(x) = \sum_{i=1}^{n} x_i^2 \text{ para} - 5,12 \leq x_i \leq 5,12$$

Onde o mínimo global é $f^*(x) = 0;\ x_i = 0,\ i = 1 : n.$

Ackleys Path $f(x) = -a \cdot \exp(-b \cdot \sqrt{\frac{1}{n}\sum x_i^2}) - \exp(\frac{1}{n}\sum \cos(c \cdot x_i)) + a + \exp(1)$

Onde $a = 20;\ b = 0,2;\ c = 2\pi;\ i = 1 : n \quad -32,768 \leq x_i \leq 32,768$ o mínimo global $f^*(x) = 0; x_i^* = 0,\ i = 1 : n.$

A função Rastring's é baseada na função de Jong's com *cosseno* adicional modulando e produzindo muitos mínimos locais. Esta função de teste é multimodal, no

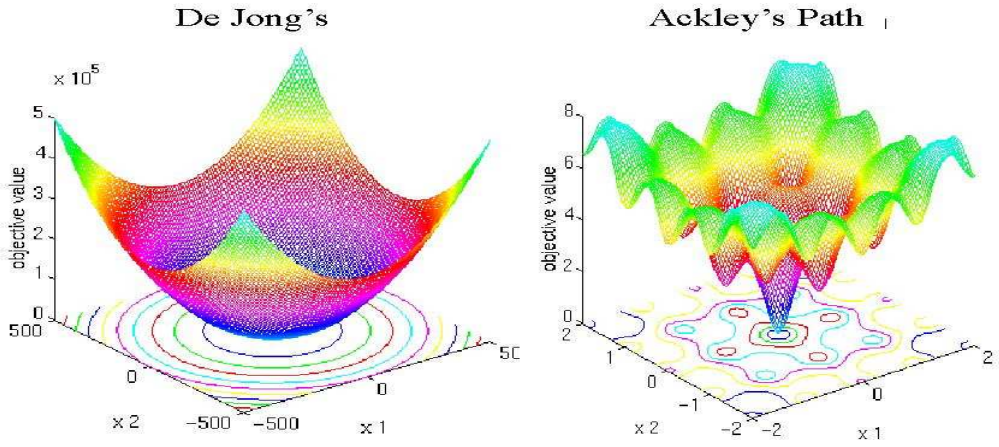

Figura 6.8: - Funções de Jong - Ackleys PATH

entanto, a localização dos mínimos são regularmente distribuidos. A função Hiper-elipsóide é contínua, convexa e unimodal.

$$\text{Rastring's } f(x) = 10 \cdot n + \sum_{i=1}^{n} (x_i^2 - 10 \cdot \cos(2\pi x_i)) \quad -5,12 \leq x_i \leq 5,12 \qquad (6.2.4)$$

o mínimo global $f(x) = 0;\ x_i = 0;\ i = 1 : n$

$$\text{Hiperelipsóide} f(x) = \sum_{i=1}^{n} \sum_{j=1}^{i} x_j^2 \quad -65,536 \leq x_i \leq 65,536$$

o mínimo global $f^*(x_i) = 0;\ x_i = 0,\ i = 1 : n$ A função de teste de Michalewicz é multimodal com $n!$ ótimos locais. Nesta função o parâmetro m define o número das regiões de vale ou extremos. Quanto maior m, maior a dificuldade da busca. Para valores de m muito grandes a função faz com que os algoritmos de busca tenham a dificuldade de como procurar uma agulha no palheiro.

$$\text{Michalewicz } f(x) = -\sum_{i=1}^{n} \text{sen}(x_i) \cdot (\text{sen}(i \cdot x_i)^2/\pi)^{2m}, \quad m = 10 \quad 0 \leq x_i \leq \pi$$

Os mínimos globais são para $f(x) = -4,687$ para $n = 5;\ x_i =$??? e para $f(x) = -9,66\ \ n = 10;\ x_i =$???.

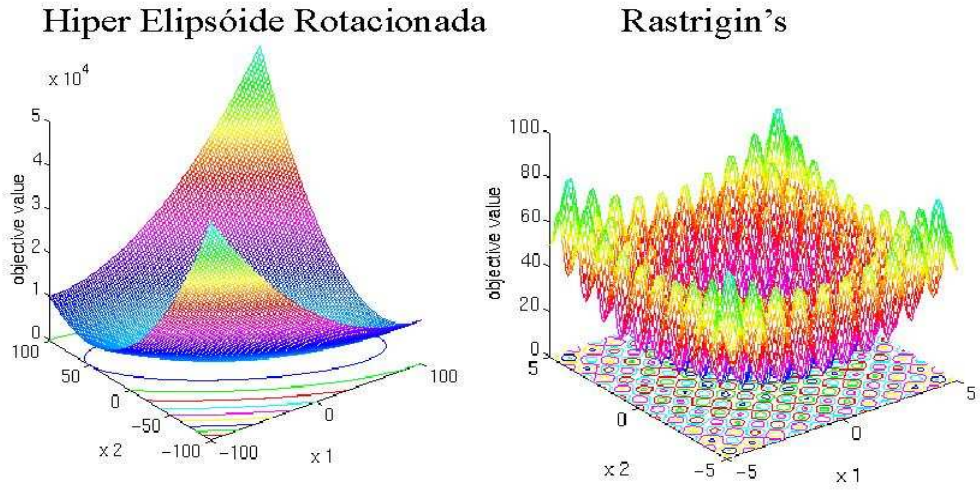

Figura 6.9: - Funções de teste Hiperelipsóide - Rastring's

Observa-se que para esta função a literatura não fornece os valores ótimos de x, apenas os valores ótimos da $f(x)$, que no teste com o TALUS também foram obtidos.

A função de Branin Rcos tem três ótimos globais e é dada pela seguinte expressão:

$$\text{Branin } f(x,y) = a(y - by^2 + cx - d)^2 + e(1 - f)\cos(x) + e$$

$a = 1,\ b = \frac{5,1}{4\pi},\ c = \frac{5}{\pi},\ d = 6,\ e = 10,\ f = \frac{1}{8\pi},\ -5 \leq x \leq 10,\ 0 \leq y \leq 15$ Os mínimos globais são: $f^*(x,y) = 0,39887;\ (x^*, y^*) = (-\pi,\ 12,275),\ (\pi,\ 2,275),\ (9,42478,\ 2,475).$

A função Goldstein-Price com um único mínimo:

Goldstein-Price $f(x,y) = [1 + (x+y+1)^2(19 - 14x + 3x^2 - 14y + 6xy + 3y^2)][30 + (2x - 3y)^2(18 - 32x + 12x^2 + 48y - 36xy + 27y^2)]$ quando $-2 \leq x,\ y \leq 2$

O mínimo global é: $f^*(x,y) = 3;\ (x,y) = (0,\ 1).$

A função *Six-Hump camel back* que possui na região de busca seis mínimos, sendo

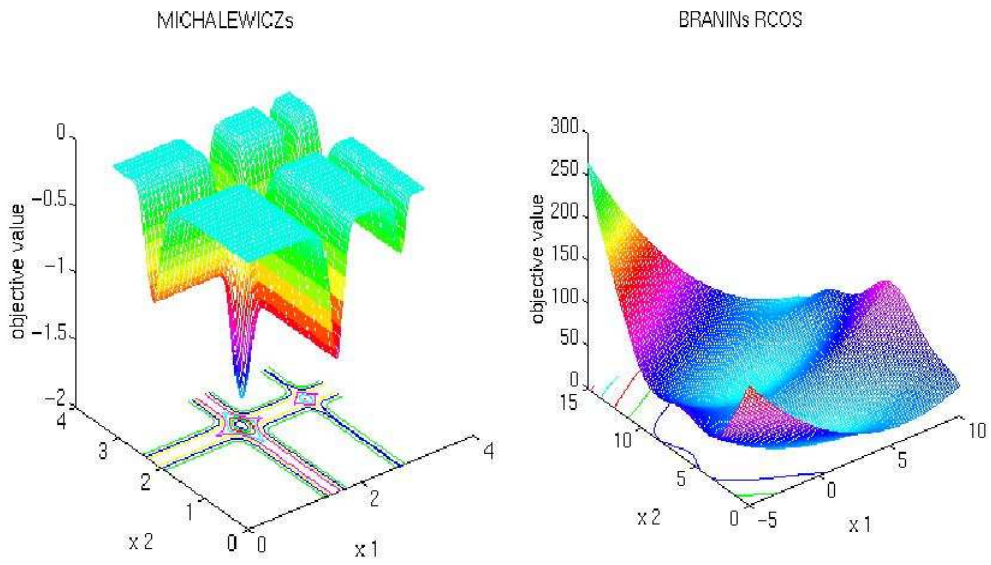

Figura 6.10: - Funções de teste Michalewicz - Branin

dois globais.

$$\text{Six-Hump } f(x) = (4 - 2,1x^2 + x^{\frac{4}{3}})x^2 + xy + (-4 + 4y^2)y^2;$$

onde $-3 \leq x \leq 3$ e $-2 \leq y \leq 2$, os mínimos globais: $f^*(x, y) = -1,0316;\ (x^*, y^*) = (-0,0898;\ 0,7126),\ (0,0898; -0,7126)$. A função de Schwefel tem uma topologia enganosa, pois, o mínimo global não segue um parâmetro de espaço em relação a um mínimo local próximo. Portanto, os algoritmos de busca são inclinados a convergirem para uma direção errada.

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

onde $-500 \leq x_i \leq 500,\ i = 1 : n$

O mínimo global: $f^*(x) = -n418,9829;\ x_i^* = 420,9687$ A função de Rosenbrock é um problema de otimização clássico, também conhecida como função Banana. O

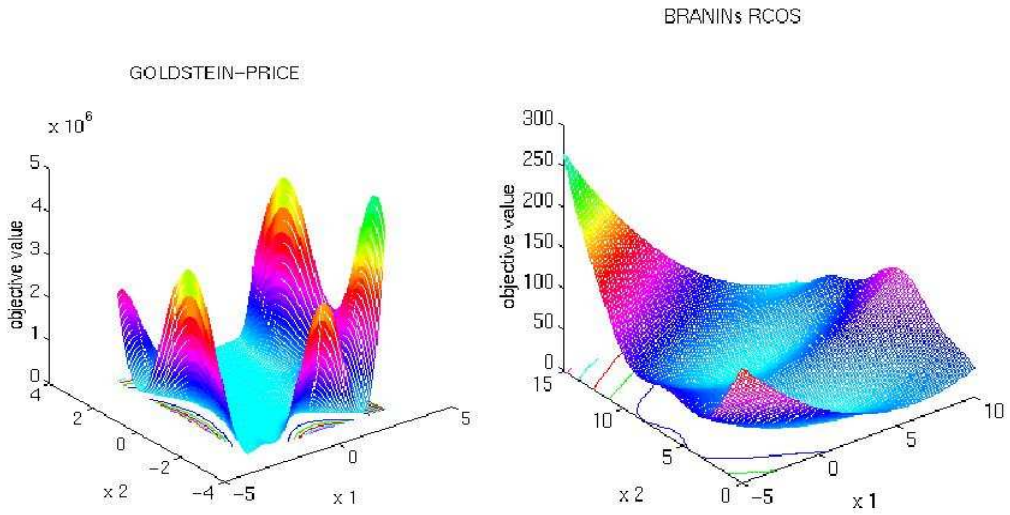

Figura 6.11: - Funções de teste Goldstein Price - Six-Hump Camel Back

ótimo encontra-se em um vale preriódico, longo, estreito, e de formato plano. Para encontrar o vale é trivial, entretanto a convergência para o ótimo global é dificultada. Esta função de teste tem sido repetidamente usada para avaliação de *performance* de algoritmos de otimização global.

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

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

O mínimo global $f^*(x) = 0;\ x_i^* = 1;\ i = 1 : n$

6.2.1 Análise Comparativa com Outros Algoritmos

No que se refere a análise de desempenho de algoritmo de otimização global existem muitas alternativas de comparação conforme descrito no Capítulo 3. Porém, a mais comum e bem aceita pela comunidade acadêmica de otimização global é realizar teste

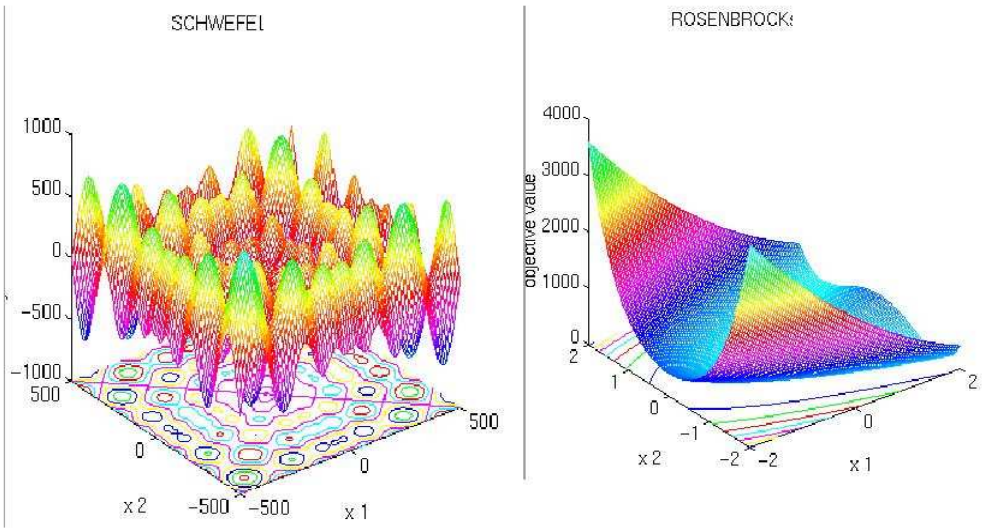

Figura 6.12: - Funções de teste Schwefel e Rosenbrock

dos algoritmos com funções com níveis de complexidade diferentes. A complexidade de uma função de teste esta associada ao número de variáveis que a compõe e a sua forma. Assim definindo, quando em uma função de teste se inclui um ruído aleatório fazendo surgir muitos ótimos locais e nenhuma regra predefine a ocorrência de ótimos globais, trata-se classifica-se esta função de alta complexidade, as funções apresentadas na seção 6.2 são comumente utilizadas por se tratarem de funções associadas a problemas de natureza prática. Em [105] apresenta um servidor avaliação automática que inclui não só a robustez e eficiência mas também a qualidade da solução. Considerando estes aspectos realiza-se a comparação conforme relatório apresentado na Tabela 6.1. Para efeito de comparação tomou-se como referência dois algoritmos livremente distribuídos o *Simulated Annealing*(ASA) e o Genético (Geno3). As funções de teste escolhidas são as de otimização irrestrita, uma vez que o TALUS ainda

não está adaptado para receber as restrições, [1].

Tabela 6.2: Análise comparativa de algoritmos de busca direta [1]

Função	dim	$f^*_{\max}$	ASA	Geno3	Talus
Jong	3	0	3,33e-06	1,78e-08	1,22e-09
Ackley	30	0	5,91e-10	2,47e-01	1,67e-02
Hiperelipsóide	30	0	4,50e-02	2,96e-02	8,6e-05
Rastring	2	0	1,60e-09	3,94e+01	6,35e-04
Michalewicz	10	-9,66	3,55e-01	1,33e+00	-9,45
Branin	2	0,39887	6,10e-05	1,20e02	3,89e-01
Goldstein-Price	2	3	0,00e+00	0,10e+00	0,3e+00
Six-hump	2	-1,0316	-1,04e-01	-4,83e+00	-1,35e+00
Schwefel	2	837,9653	5,26e+10	8,76e+05	9,63e+02
Rosenbrock	10	0	2,86e-03	7,89e-03	1,35e-03

Na Tabela 6.2 comparando-se os resultados do TALUS com a coluna $f^*_{\max}$ que são os valores ótimos teóricos, nos diversos testes realizados contata-se que o TALUS apresenta um melhor desempenho em relação aos demais algoritmos testados. Considere-se que o número de iterações aqui é de 312, não se fazendo mensão ao tamanho da população inicial para se obter amostras da função objetivo. Para efeito do teste toma-se 200 pontos, em cada iteração, do espaço de busca.

O TALUS gera uma seqüência $\{f(x^k_{\max})\}$ que é monotônica não decrescente, sendo os seus elementos definidos num conjunto compacto, que é subconjunto do contradomínio f. Esta compacticidade é garantida pela hipótese da existência máximo. Uma situação que não satisfaz a essas condições é apresentada no exemplo abaixo:

$$f(x) = \begin{cases} x^2, \text{ se } \quad 0 \leq x \leq 1; \\ \\ 0, \text{ se } \quad x = 1; \\ \\ (x-2)^2, \text{ se } \quad 1 < x \leq 2 \end{cases}$$

Observa-se que embora o problema não atenda a condição de compacticidade do subconjunto do contradomínio de f, ainda assim o TALUS é capaz de encontrar os

pontos extremos desta função no intrvalo dado, pois, no ponto em que a função f não apresenta continuidade a probabilidade de convergência para aquele ponto é mínima. Esse comportamento faz com que o TALUS vá buscar os pontos na proximidade do extremo, pois, o $f^k_{\max}$ não será substituído enqunato não houver um $f^{k+1}_{\max}$ maior.

6.2.2 Algumas Classes de Problemas Solucionadas pelo Talus

Programação Não-Linear, Não-Convexa com Restrições

No problema de programação não-linear com restrição a seguir, a função objetivo é não côncava e o conjunto viável é não convexo. Portanto, as condições de Karush-Kuhn-Tucker não são condições de necessidade.

$$\begin{aligned} \max_{x,y} \quad & f(x,y) = x^2 - y^2; \\ x - \frac{16}{25}(y-1)^2 - 4 &\leq 0; \\ x - (y-2)^2 - 4 &\leq 0; \\ -x + 8y - 28 &\geq 0; \\ x_1 \geq 0, \quad & x_2 \geq 0. \end{aligned}$$

A solução obtida é dada pelo ponto $(x, y) = (20, 6)$. É o ponto de tangência da reta de restrição à uma das parábolas de restrição. Neste ponto o conjunto viável forma um "bico"; uma ponta, conforme Figura 6.13.

Para se obter a solução utilizando o Talus é necessário fazer uma transformação do problema com restrição para irrestrito. Esta transformação pode ser obtida utilizando o método das penalidades. Se um ponto da nuvem encontrar-se fora do conjunto viável, deve-se penalizar fortemente a função objetivo. Obtém-se então uma função transformada dada por:

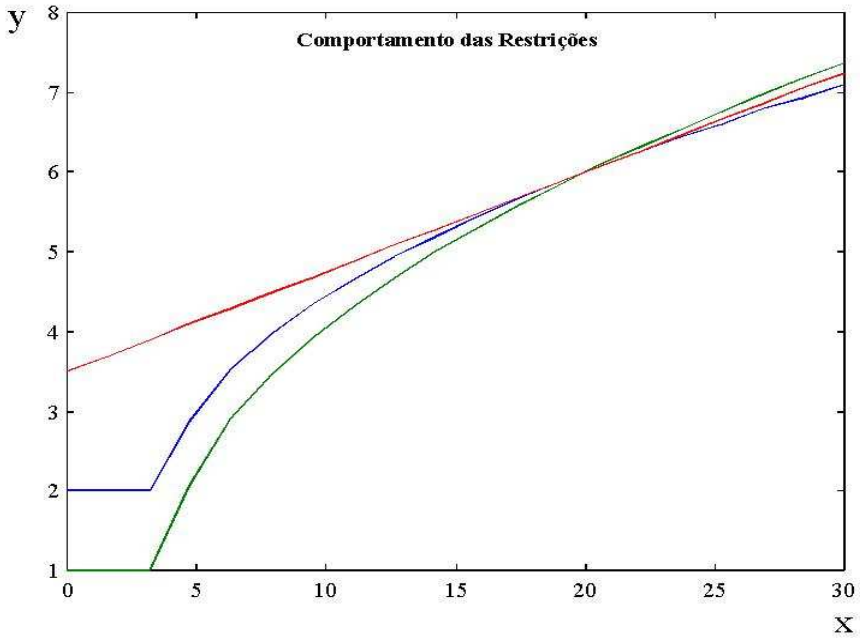

Figura 6.13: Visualização do conjunto viável.

$$
\begin{aligned}
f_T(x,y) &= f(x,y) + \text{ penalidades} \\
&= f(x,y) + P_1(x,y) + P_2(x,y) + P_3(x,y) = \\
&= x^2 - y^2 + \\
&+ \begin{cases} \text{se} \quad x - \frac{16}{25}(y-1)^2 - 4 \leq 0, \quad \text{faça} \quad P_1(x,y) = 0; \\ \text{caso contrário,} \qquad\qquad\quad \text{faça} \quad P_1(x,y) = -M \end{cases} + \\
&+ \begin{cases} \text{se} \quad x - (y-2)^2 - 4 \leq 0, \quad \text{faça} \quad P_1(x,y) = 0; \\ \text{caso contrário,} \qquad\qquad\quad \text{faça} \quad P_1(x,y) = -M \end{cases} + \\
&+ \begin{cases} \text{se} \quad -x + 8y - 28 \geq 0, \quad \text{faça} \quad P_1(x,y) = 0; \\ \text{caso contrário,} \qquad\qquad\quad \text{faça} \quad P_1(x,y) = -M, \end{cases}
\end{aligned}
$$

onde $M > 0$ é a penalização.

O espaço de busca adotado para o exemplo é: $[0, 50] \times [0, 50]$ e o número $M = 10.000$.

O Problema do Cálculo das Variações

O problema clássico do cálculo das variações é:

$$\underset{y}{\text{Max}}\, J = \int_{t_0}^{t_1} I(y(t), \dot{y}(t))dt$$

$$y(t_0) = y_0;$$

$$y(t_1) = y_1.$$

Trata-se de escolher uma função que maximiza um funcional. Considere-se, para simplificar a exposição, o caso invariante no tempo. Tome-se uma base completa de funções $\{g_i\}$ de $\mathbb{R}$ em $\mathbb{R}$. Desta forma pode-se escrever

$$y(t) = x_1 g_1(t) + x_2 g_2(t) + x_3 g_3(t) + \cdots + x_q g_q(t),$$

onde os x's são os coeficientes da combinação linear dos elementos da base do espaço vetorial de funções. A combinação linear será truncada num termo q que será escolhido em função da aproximação desejada.

Esses x's corresponderão às coordenadas de um vetor que será um ponto i da nuvem do algoritmo TALUS. Cada ponto da nuvem será uma função y^i dada por:

$$y^i = x_1^i g_1(t) + x_2^i g_2(t) + x_3^i g_3(t) + \cdots + x_q^i g_q(t).$$

No espaço euclideano, de ordem q, onde trabalha o TALUS, esse ponto da nuvem de funções corresponde a um ponto do espaço euclideano $\mathbb{R}^q$; um vetor do $\mathbb{R}^q$:

$$x^i = \begin{bmatrix} x_1^i \\ x_2^i \\ \vdots \\ x_q^i \end{bmatrix}$$

Obtendo-se:

$$\dot{y^i}(t) = \frac{dy^i}{dt} = x_1^i \frac{dg_1}{dt} + x_2^i \frac{dg_2}{dt} + \cdots + x_q^i \frac{dg_q}{dt}.$$

Uma vez gerada a nuvem, cada um de seus pontos serão números reais. As expressões de $y^i(t)$ e $\dot{y}(t)$ são então substituídas na expressão de $I(y(t), \dot{y}(t))$, e o valor de J é calculado (pode-se e deve-se usar o método de Monte Carlo para se calcular a integral, usando os próprios recursos do *software*). Desta forma o problema do cálculo das variações foi transformado num problema de programação global, que é:

$$\underset{x_1, x_2, \ldots x_q}{\text{Max}} \; J(x_1, x_2, \ldots x_q),$$

onde a função J é dada de uma forma implícita.

Em cada passo há que se fazer uma integração numérica e portanto as exigências computacionais são maiores.

Equações Diferenciais

Uma outra classe de problemas solucionáveis pelo TALUS são as equações diferenciais. Considere-se a equação diferencial:

$$\frac{dy}{dt} = f(y),$$

onde $y(t) \in \mathbb{R}^n$. A idéia segue as mesmas linhas do problema do cálculo das variações. A função a ser minimizada aqui, no problema de otimização global, é:

$$F(y) = \left[\frac{dy}{dt} - f(y)\right]^T \left[\frac{dy}{dt} - f(y)\right],$$

sendo y, e portanto F, uma função de $x_1, x_2, \ldots x_q$.

No caso da equação de van der Pol, por exemplo,

$$\frac{d^2y}{dt^2} + 2\xi\omega_0(\beta y^2 - 1)\frac{dy}{dt} + \omega_0^2 y = 0,$$

que não tem solução analítica fechada, pode-se tomar uma base de funções sinusoidais (como na série de Fourier), pois sabe-se que ela tem solução periódica (é a solução em regime permanente).

As condições iniciais, de uma maneira geral, entram como restrições nos x's. Neste caso, cai-se num problema de programação não-linear (global), com restrições, que, como já se viu, pode ser resolvido pelo TALUS.

Capítulo 7

RESULTADOS, CONCLUSÕES E RECOMENDAÇÕES PARA TRABALHOS FUTUROS

7.1 Resultados Gerais e Conclusões

Existe um grande interêsse em se pesquisar algoritmos de busca direta. Embora abrangente a pesquisa ainda apresenta grandes dificuldades na obtenção de bons resultados que atendam a classes mais amplas de problemas de OG. Em muitos estudos, quando observadas as contribuições para melhorar a eficiência na obtenção de uma boa direção de busca do ótimo, identificam-se muitos tipos de fraquezas metodológicas, particularmente pela falta de hipóteses estruturadas e gerais. Tem sido observado que muitas das novas pesquisas buscam apoio em heurísticas levando a simplificações na busca da identificação de aspectos mais formais. O presente trabalho explora as estruturas comuns dos algoritmos de busca direta, tanto os probabilísticos quanto os

que usam heurísticas, descritos nos Capítulos 3 e 4.

É relativamente recente os desenvolvimentos em algoritmos de OG de busca direta, que têm sido explorados principalmente em aplicações práticas nas diversas áreas da Engenharia. O objetivo geral é preencher os vazios relativos à determinação de um procedimento eficiente da direção de busca e critérios de convergência mais gerais. Esta Tese investiga e desenvolve procedimentos de OG com a obtenção da direção de busca do ótimo de forma eficiente, trazendo contribuíções sobre algoritmos baseados na busca direta e estabelecendo procedimentos básicos para desenvolvimento e análise de algoritmos de OG.

A construção de um algoritmo de OG conduz ao estabelecimento de uma estrutura básica que seja aderente ao problema de tomada de decisão, incorporando os contextos; variáveis típicas, objetivos e estruturas básicas de ações. Então, o procedimento de OG inicia a assistência aos modelos de decisão.

Outro aspecto observado na literatura sobre OG é a ênfase nas medidas de desempenho dos algoritmos e praticidade. Neste aspecto, a busca de algoritmos de uso em uma ampla classe de problemas de OG é perseguida, embora muitas vezes comprometendo o desempenho.

Este trabalho desenvolve contribuições no desenvolvimento de busca de algoritmos de OG de busca direta que atendam um número o mais amplo possível de classes de problemas.

As principais características desenvolvidas neste trabalho são relacionadas a seguir:

- Inclusão da formulação da estrutura dos algoritmos de busca direta mais gerais, identificando: os procedimentos da busca, as estratégias definidas para a busca e critério de convergência. Identificam-se também as classes de problemas que

estes algoritmos atendem.

- Análise analítica da convergência do TALUS considerando-se qualquer classe de funções.

- Avaliação dos parâmetros internos do TALUS e seus efeitos no desempenho (Tempo de Resposta X Precisão).

- Análise de aplicações mais gerais de OG, ampliando a área de abrangência (problemas de cálculo das variações, equações diferenciais).

- Desenvolvimento de um *software* de OG capaz de otimizar qualquer função objetivo.

- Aplicações e análise comparativa com outros *softwares* utilizando funções de teste tidas como *benchmark* pela comunidade de OG.

As hipóteses adotadas neste trabalho dão suporte para, através do algoritmo TALUS, resolver problemas de programação inteira e combinatória. Resultados podem ser obtidos através de pequenas modificações no algoritmo. Deve-se ressaltar a fácil aplicação do algoritmo e o alto desempenho obtidos através do *software* apresentado.

7.2 Recomendações e Trabalhos Futuros

Este trabalho é voltado para o desenvolvimento e análise de algoritmos de OG que podem ser utilizados em classes mais gerais ou particulares de problemas. Através da definição e formulação do problema os passos associados à estratégia de solução devem ser definidos em combinação com os aspectos analíticos do problema.

As aplicações nas áreas de otimização de Redes Neurais, combinatória - aplicações de otimização de áreas de disco e memória, alocação de recursos, são necessidades imediatas, pois, embora os algoritmos atuais tenham sido eficientes, ainda são muito lentos.

Na versão atual o TALUS é *software* para fins didáticos devendo ser desenvolvido para uma versão de suporte à atividade industrial que demande problemas de dimensão muito elevada com um número elevado de restrições.

Podem ser realizadas avaliações da utilização do TALUS como um acelerador algoritmo genético, fazendo testes com estratégias híbridas, aproveitando que o TALUS converge rapidamente para uma região próxima do ótimo global e o algoritmo genético tem por característica uma evolução mais lenta.

Bibliografia

[1] Univie, "Site on global optimization," *www.mat.univie.ac.at/~neum/glopt.html*, vol. last visit april, 2004.

[2] G. D. LINS, "Estudo, melhorias e implementação computacional do Talus," Dissertação de Mestrado, Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Pernambuco, Recife, Pernambuco, Março 1999, orientador: Prof. Fernando Menezes Campello de Souza.

[3] F. M. C. de Souza, *Decisões Racionais em Situações de Incerteza*, UFPE, Ed. Recife, Brasil: Editora Universitária UFPE, 2002.

[4] J. Maciel, *Elementos de teoria geral dos sistemas*, Vozes, Ed. Editora Vozes, Petrópolis - 1974.

[5] A. M. Cavalcanti and C. Mariano, "Controle de tráfego telefônico usando base de conhecimentos," *XXXV SBPO*, vol. ISSN 1518-1731, Novembro- Natal RN 2003.

[6] A. M. Cavalcanti, "Controle de sobrecarga em centrais de telefonia tipo CPAT," *XXVIII SBPO*, vol. ISSN 1518-1731, Outubro - Vitória ES 1997.

[7] A. M. Cavalcanti, J. R. R. Cavalcante, and F. M. C. de Souza, "Análise de previsão de vendas em uma empresa de telecomunicações," *IV Congresso Brasileiro de desenvolvimento de produtos*, Set Gramado RS 2003.

[8] A. M. Cavalcanti and J. R. R. Cavalcante, "Procedimentos para análise de previsão," *XXXV Simpósio Brasileiro de Pesquisa Operacional*, Nov Natal RN 2003.

[9] A. M. Cavalcanti, J. R. Cavalcante, C. A. V. Cavalcante, and et al, "Optimal replacement policy using optimization stochastic method," *IEEE tansations Reliability, submited 2004/june*, june 2004.

[10] R. Horst and P. M. Pardalos, *Handbook of global optimization.* Kluwer Academic Publishers, ISBN 0-7923-3120-6, 1995.

[11] S. Voss, S. S. Martello, I. H. Osman, and C. Roucariol, *Meta-Heuristics: advances and trends in local search paradigms for optimization*, K. A. Publishers, Ed., 1999.

[12] V. V. Moiseenko and V. V. Yatskevisch, "Global optimization based on heuristic self-organization: The discrete case," *Cybernetics and Systems Analysis*, vol. 37, pp. 727–734, Sep/Out 2001.

[13] R. Fourer, "Nonlinear programming frequently asked questions." *Optimization Technology Center of Northwestern University and Argonne National Laboratory*, p. last access 29/04/2004. [Online]. Available: http://www-unix.mcs.anl.gov/otc/Guide/faq/nonlinear-programming-faq.htm%l

[14] R. Hooke and T. A. Jeemes, "Direct search solution of nand statistical problems," *Journal ACM*, vol. 8, pp. 212–229, 1961.

[15] C. Khompatraporn, Z. B. Zabinsky, and J. D. Pintér, "Comparative assessment of algorithms and software for global optimization," *Submitted for publication*, 2001.

[16] N. Metropolis, A. W. Rosenbluth, N. M. Rosenbluth, A. H. Teller, and E. Teller, "Equations of state calculations by fast computing machines," *The Journal of Chemical Physics*, vol. 21:1087-1092, 1953.

[17] J. R. R. Cavalcante, "Algoritmo paralelo para otimização global," Master Thesis, Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Pernambuco, Recife, Pernambuco, September 1996, chairman: Prof. Fernando Menezes Campello de Souza.

[18] J. G. Belém, "Proposta de circuitos eletrônicos para implementação de algoritmos probabilísticos," Dissertação de Mestrado, Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Pernambuco, Recife, Pernambuco, Agosto 1999, orientador: Prof. Fernando Menezes Campello de Souza.

[19] A. M. Cavalcanti, J. R. R. Cavalcante, and F. M. C. de Souza, "Stochastic global optimization: a new probabilistic algorithm - talus," *IEEE tansations on systems, Man and Cybernetics*, Submeted june 2004.

[20] D. M. Himmelblau, *Applied nonlinear programming.* McGroaw-Hill, 498 p., 1972.

[21] J. J. Moré and S. J. Wright, *Optimization software guide*. Boston, USA: SIAM Frontiers in applied Mathematics Series, vol 14 1993.

[22] M. Eigen and R. Winkler, *Das spiel*. Piper & Co., Müchen Germany, 1975.

[23] A. Schrijver, *Theory oLinear and Integer Programming*. New York, USA: Jonh Willey Sons, 1986.

[24] W. Rudin, *Principles of mathematical analysis*, M. Hill, Ed. McGrow Hill, 1976.

[25] M. R. Osborne, *Algorithms in optimization and data analysis*. New Jersey, USA: John Willey and Sons, 1985.

[26] R. T. Rockfellar, *Convex analysis*. New Jessey, USA: Princiton University Press, 1970.

[27] G. L. Torres, *Métodos práticos de otimização*, UFPE, Ed. Notas de aula curso de Eng. Elétrica, 2000, notas de Aula UFPE - DES.

[28] R. B. et al, *Álgebra linear*. Harper Row, Brasil, 1980.

[29] G. Strang, *Linear algebra and its applications*. Orlando, USA: Harcourt Brace company, 1988.

[30] A. R. Conn, N. I. M. Gould, and P. L. Toint, *Integral global optimization, Lecture Notes in Economics and Mathematical Systems*, C. Press, Ed. Oxford: I. S. Duff and G. A. Watson, 1997, pp 363-390.

[31] D. F. Shanno and E. M. Simantiraki, *Integral global optimization, Lecture Notes in Economics and Mathematical Systems*, C. Press, Ed. Oxford: I. S. Duff and G. A. Watson, pp 339-362, 1997.

[32] A. Strekalovisk, "On search of global maximum of convex functions on constrint set," 1997.

[33] S. H. Chew and Q. Zheng, *Integral Global Optimization, Lecture Notes in Economics and Mathematical Systems*, S. Verlag, Ed. John Wiley & Sons, Vol 298, 1988.

[34] B. Calvert and M. K. Vamanamurthy, "Local and global extrema for functions of several variables," *J. Austral. Amth. Soc.(series A)29,362-368*, 1980.

[35] S. Lucide and M. Sciandrone, "On the global convergence of derivative-free methods for unconstrained optimization," *SIAM Journal Optim*, vol. 18, pp. 97–116, June 2002.

[36] L. Grippo, F. Lampariello, and S. Lucidi, "Global convergence and stabilization of unconstrained minimization methods without derivation," *J. Optim Theory appl*, vol. 56, pp. 385–406, 1988.

[37] V. Torczon, "On the convergence of pattern search algorithms," *SIAM J. Optim*, vol. 7, pp. 1–25, 1997.

[38] H. J. Kushner, "A versatile stochastic model of a function of unknown and time-varying form," *Journal of Mathematical Analysis and Applications*, vol. 5, pp. 150–167, 1962.

[39] Hiriart-Urruty., *Conditions for global otimality*, K. A. Publishers, Ed. Kluwer Academic Publishers, 1995.

[40] M. Lewis and V. Torczon, "Direct search: then and now," *Journal Comput. Appl. Math.*, vol. 124, pp. 191–207, 2000.

[41] B. B. Mandelbrot, *The fractal geometry of nature.* Freeman & Co., New York USA, 1983.

[42] J. D. Murray, *Mathematical biology.* Springer-Verlag., Berlin Germany, 1989.

[43] J. L. Casti, *Searching for certainty*, M. . Co., Ed. Morrow e Co, New York, 1990.

[44] I. Stewart, *Nature's numbers.* Harps and Collins, New York, 1995.

[45] S. Wolfram, *The mathematica book.* Wolfram Media and Cambridge, 1996.

[46] C. Jacob, *Illustrating evolutionary computation with mathematica.* Morgan Kaufmann, San Francisco USA, 2001.

[47] T. G. Kolda, R. M. Lewis, and V. Torczon, "Optimization by direct search: new perspectives on some classical and modern methods," *SIAM Review*, vol. 45, pp. 382–482, September 2003.

[48] A. A. Zhigljavsky, *Theory of global randon search*, K. A. Publishers, Ed., 1991.

[49] C. G. E. BOENDER and H. E. ROMEIJN, *HandBook of glabal optimization.* Kluwer Academic Publishers, 1995, ch. Stochastic Methods, pp. 829–869.

[50] J. D. Pintér, *Global optimization in action (continuos and Lipschitz optimization: algorithms, implematation and applications.* Boston, USA: Kluwer Academic Publishers, 1996.

[51] E. W. J. Mockus, A. Mockus, L. Mockus, and G. Reklaitis, *Bayesian heuristic approach to discrete and global optimization*, K. A. Publishers, Ed., 1996.

[52] A. Neumaier, *Interval methods for systems of equatins*, U. Press, Ed. Cambridge University, 1990, cambridge.

[53] E. R. Hansen, *Global optimizatiom using interval analysis*, M. Dekker, Ed. Macel Daker, New York, 1992.

[54] R. Horst and H. Tuy, *Global optimization - deterministic aproaches*, Heidelberg, Ed. Springer, New York, 1996.

[55] R. B. Kearfott, *Rigorous global search: continuos problems*, K. A. Publisher, Ed. Kluwer Academic, New York, 1996.

[56] J. Pintér, *Continuos global optimization software: a brief review.* Boston, USA: Optima, 52:1-8 WEB version is available at: http://plato.la.asu.edu/gom.html., 1996.

[57] C. A. Floudas, *Deterministic global optimization: theory, algorithms and application*, K. A. Publisher, Ed. Kluwer Academic Publishers, Boston, 1999.

[58] R. G. S. e Y. D. Sergeyev, *Global optimization with non-convex constraints: sequential and parallel algorithms*, K. A. Publisher, Ed. Kluwer Academic Publishers, London 2000.

[59] I. Diener, *HandBook of glabal optimization.* Kluwer Academic Publishers, 1995, ch. Trajetory methods in global optimization, pp. 649–668.

[60] W. Forster, *HandBook of glabal optimization.* Kluwer Academic Publishers, 1995, ch. Homotopy methods, pp. 669–750.

[61] Q. Zheng and D. Zhuang, "Integral global minimization: algorithms, implementations and numerical tests." *Journal Global Optimization*, vol. 7, pp. 421–454, 1995.

[62] J. Hichert, A. Hoffman, and H. X. Phú, *Developments in glabal optimization.* Kluwer Academic Publishers, 1997, ch. Convergence speed of an integral method for computing the essential supremum, pp. 153–170.

[63] H. P. Benson., *HandBook of glabal optimization.* Kluwer Academic Publishers, 1995, ch. Concave minimization: theory, applications, and algorithms, pp. 43–148.

[64] K. A. Dill, A. T. Phillips, and J. B. Rosen, *Developments in glabal optimization.* Kluwer Academic Publishers, 1997, ch. Molecular struture prediction by global optimization, pp. 217–234.

[65] J. J. Moré and Z. Wu, "Global continuation for distance geometry problems," *SIAM Journal on Optimization*, 7:814-836 1997.

[66] F. Glover and M. Laguna, *Tabu search.* Kluwer Academic, London, 1997.

[67] A. V. Levy and S. Gomez, "The tunneling method applied for the global minimization of functions," *SIAM Journal on Optimization*, 213-244 1985.

[68] I. H. Osman and J. P. Kelly, *Meta-Heurisitics: theory and applications.* Kluwer Academic, London, 1996.

[69] C. Khompatraporn, Z. B. Zambinsky, and J. D. Pintér, "Comparative assessment of algorithms and software for global optimization," *Submited for publication*, 2001.

[70] A. Neumaier, "Global optimization," vol. last access 29/02/2004. [Online]. Available: http://www.mat.univie.ac.at/~neum/glopt.html

[71] H. D. Mittelmann and P. Spellucci, "Decision tree for optimization software," *last access 29/02/2004 - http://plato.la.asu.edu/guide.html*, 2001.

[72] R. Fourer, D. M. Gay, and B. W. Kernighan, *AMPL- A modeling launguage for methematical programming*, T. S. Press, Ed. Boyd and Fraser, Massachusetts, 1990.

[73] A. Brooke, D. Kendrick, and A. Meerans, *GAMS: A user's guide.* Redwood City, USA: Cientific press, 1988.

[74] S. LINDO, *Solver suite LINDO systems.* Chicago, USA: LINDO Systems inc., 1996.

[75] M. system, *Maximal software.* Arlington, USA: Maximal software Inc., 1998.

[76] F. S. Inc., *Premium solver plataform - field - installable solver engines - User guide.* Nevada, USA: Frontline System Inc., 2001.

[77] J. J. Moré, B. S. Garbow, and K. E. Hillströn, "Testing uncontrained optimization software," *ACM transations on mathematical software*, 7:17-41 1981.

[78] W. H. e K. Schittowski, *Test examples for nonlinear programming codes*, Springer-Verlang., Ed. volume 187 of Lecture Notes in economics and mathematical systems, Berlim 1981.

[79] L. D. e G. P. Szegö, *Towards Global Optmization*, T. netherlands, Ed. Volumes 1-2, Amsterdam 1978.

[80] C. A. F. e P. M. Pardalos, *A collection of test problems for constrained global optimizations algorithms*, Springer-Verlag, Ed. Volumes 455 of lecture notes in computer science, Berlin 1990.

[81] C. J. e O. Knüppel, *A global minimization method: The multi-dimensional case*, T. FIUK, Ed. Bricht 92.1, Harburg Germany 1992.

[82] H. D. Mittelman and P. Spellucci, "Decision tree for optimization," vol. last access 14/mar/2004. [Online]. Available: http://plato.la.asu.edu/guide.html

[83] A. R. Colville, "A comparative study of nonlinear programming codes," *Technical report IBM- 320-2949*, New York, 1968.

[84] E. D. Eason and R. G. Fenton, "A comparison of numerical optimization methods for engineering design," *ASME J. Eng. Ind. Ser. B*, vol. 96(1), pp. 196–200, 1974.

[85] K. Schittkowski, *Nonlinear programming codes: information, tests, performance*, S. Verlag, Ed. Lectures notes in economics and mathematical systems, 1980, v183 - New York.

[86] M. B. Beck, *Water quality management: a review of the devolopmente and application of mathematical models*, S. Verlag, Ed. Lectures notes in economics and mathematical systems, New York, 1985.

[87] E. M. T. Hendrix, *Optimization global at work*, U. Wageningen, Ed., PhD Thesis, agricultural University Wageningen, 1998.

[88] C. J. P. Bélisle, "Convergence theorems for a class of simulated annealing algorithms on R^4," *Journal of Applied Probability, 29(4):885-895,1995*, 1992.

[89] P. Svenson and H. Sidenbladh, "Determining possible avenues of approach using ANTS," *http://www.foi.se/fusion/*, nlin.AO/0304006v1 2003.

[90] T. M. Cover and J. A. Thomas, *Elements of information theory.* Boston, USA: John Willey and Sons, 1991.

[91] D. E. Goldeberg, *Genetic algorithms in aearch, aptimization and machine learning.* MA, USA: Addison-Wesley, 1989.

[92] K. A. de Jong, *An analysis of the behaviour of a class of genetic adaptive systems.* Michigan, USA: Ph.D Thesis University of Michigan, Ann Arbor, MI, 1975.

[93] D. Vose and G. E. Liepins, "Punctuade equilibria in genetic search," *Complex system*, vol. 5:31-44, 1991.

[94] A. Nix and M. D. Vose, "Modelling genetic algorithms with Markov Chains," *Ann. Math - Artificial Intelligence*, vol. 5:439-459, 1992.

[95] S. Aytug and G. P. B. Koehler, "A markov chain analysis of genetic algorithms with power of 2 cardinality alphabets," *European J. Oper. Res*, vol. 96:195-201, 1996.

[96] D. Greenhalgh and S. Marshall, "Convergence criteria for genetic algorithms," *SIAM J. Comput.*, vol. 30:1:269-282, 2000.

[97] M. Lewis and V. Torczon, "Ranck ordinering and positive bases in pattern search algorithms," *Technical Report TR 96-71, ICASE, NASA Langley Research Center*, vol. TR 96-71, pp. 96–71, 1996.

[98] C. Audet and J. E. D. Júnior, "Analysis of generalized pattern searches," *SIAM J. Optim*, vol. 13, pp. 889–903, 2003.

[99] B. E. Stuckman and E. E. Easom, "A comparation of bayesian sampling global optimization techniques," *IEEE tansations on systems, Man and Cybernetics*, vol. 22, pp. 1024–1032, september/october 1992.

[100] Y. Q. Duan, V. K. Gupta, and S. Sorooshian., *Shuffled complex evolution approach for efficiente global minimization.* Journal of Optimization Theory and Application, Vol. 76, No. 3, pp 501-521, 1993, march.

[101] B. C. Cetin, J. Barhen, and J. W. Burdich, *Terminal repeller unconstrained sugenergy tunneling for fast global optimization.* Journal of Optimization Theory and Applications, Vol 77, No.1, pp 97-126, 1995, april.

[102] M. H. Haussoun, *Fundamentals of artificial neural networks*, A. B. Book, Ed. The MIT Press, 1995, 511p.

[103] D. Greenhalgh and S. Marshall, "Convergence criteria for genetic algorithms," *SIAM Journal Comput*, vol. 30, pp. 269–282, May 2000.

[104] R. J. de Araújo, "Confiabilidade de *software* como um parâmetro de qualidade," Monografia - Bacharelado, Programa de Graduação em Bacharelado em Sistemas de Informação, Recife, Pernambuco, Novembro 2003- Faculdade Integrada do Recife - FIR, orientador: André Marques Cavalcanti.

[105] H. D. Mittelmann and A. Prussne, *A server for automated performance analysis and benchmarketing of optimization software*, A. B. Book, Ed. Kluwer Academic Publishers, 2003, vol. Handbook of Global Optimization V2.