3
Algoritmos Evolucionários

3.1
Algoritmos Genéticos

Essencialmente, Algoritmos Genéticos são métodos de busca e otimização que tem sua inspiração nos conceitos da teoria de seleção natural das espécies proposta por Darwin (Goldberg, 1989) (Koza, 1992) (Mitchell, 1994) (Back, 1996) (Fogel et al, 1966).

Os sistemas desenvolvidos a partir deste princípio são utilizados para procurar soluções de problemas complexos ou com espaço de soluções muito grande (espaço de busca), o que os tornam problemas de difícil modelagem e solução quando se aplicam métodos de otimização convencionais.

Estes algoritmos são inspirados nos processos genéticos de organismos biológicos para procurar soluções ótimas ou sub-ótimas. Para tanto, procede-se da seguinte maneira: codifica-se cada possível solução de um problema em uma estrutura chamada de "cromossomo", que é composta por uma cadeia de bits ou símbolos. Estes cromossomos representam indivíduos, que são evoluídos ao longo de várias gerações, de forma similar aos seres vivos, de acordo com os princípios de seleção natural e sobrevivência dos mais aptos, descritos pela primeira vez por Charles Darwin em seu livro "A Origem das Espécies". Emulando estes processos, os algoritmos genéticos são capazes de "evoluir" soluções de problemas do mundo real.

Os cromossomos, ou indivíduos, são então submetidos a um processo evolucionário que envolve avaliação, seleção, recombinação (*crossover*) e mutação. Após vários ciclos de evolução a população deverá conter indivíduos mais aptos. Os algoritmos genéticos utilizam uma analogia direta deste fenômeno de evolução na natureza, onde cada indivíduo representa uma possível solução para um problema dado. A cada indivíduo atribui-se um valor de avaliação: sua aptidão, que indica quanto a solução representada por este indivíduo é boa em relação às outras soluções da população. Desta maneira, o termo *População* refere-se ao conjunto de todas as soluções com as quais trabalha o sistema. Aos indivíduos mais adaptados é dada uma probabilidade maior de se reproduzirem mediante cruzamentos com outros indivíduos da população, produzindo

descendentes com características de ambas as partes. A mutação também tem um papel significativo, ao introduzir na população novos indivíduos gerados de maneira aleatória.

O processo de evolução começa com a criação aleatória dos indivíduos que formarão a população inicial. A partir de um processo de seleção baseado na aptidão de cada indivíduo, são escolhidos indivíduos para a fase de reprodução que cria novas soluções utilizando-se, para isto, um conjunto de operadores genéticos. Deste modo, a aptidão do indivíduo determina o seu grau de sobrevivência e, assim, a possibilidade de que o cromossomo possa fazer parte das gerações seguintes.

O procedimento básico de um algoritmo genético é resumido na Figura 4 (Davis, 1996).


```
Inicio
t←1
inicializar população P(t)
avaliar população P(t)
enquanto (não condição_de_fim) faça
t←t+1
selecionar população P(t) a partir de P(t-1)
aplicar operadores genéticos
avaliar população P(t)
fim enquanto
fim
```

Figura 4 – Procedimento básico do algoritmo genético

Para determinar o final da evolução pode-se fixar o número de gerações, o número de indivíduos criados, ou ainda condicionar o algoritmo à obtenção de uma solução satisfatória, isto é, quando atingir um ponto ótimo. Outras condições para a parada incluem o tempo de processamento e o grau de similaridade entre os elementos numa população (convergência).

As seções seguintes apresentam em mais detalhes cada um dos componentes de um algoritmo genético.

3.1.1
Representação

A representação é um aspecto fundamental na modelagem de um algoritmo genético para a solução de um problema. Neste estágio define-se a estrutura do cromossomo, com os respectivos genes que o compõem, de maneira que este seja capaz de descrever todo o espaço de busca relevante do problema. Os principais tipos de representação são: binária, números reais, inteiros, grupamento de inteiros e baseadas em ordem.

3.1.2
Codificação e Decodificação

A solução de um problema pode ser representada por um conjunto de parâmetros (genes), unidos para formar uma cadeia de valores (cromossomo); a este processo chama-se codificação. As soluções (cromossomos) são codificadas através de uma seqüência formada por símbolos de um sistema alfabético. Originalmente, utilizou-se o alfabeto binário (0, 1), porém, novos modelos de AGs codificam as soluções com outros alfabetos, como, por exemplo, com números reais (Michalevicz, 1996).

A decodificação do cromossomo consiste basicamente na construção da solução real do problema a partir do cromossomo. O processo de decodificação constrói a solução para que esta seja avaliada pelo problema.

3.1.3
Avaliação

A avaliação permite ao algoritmo genético determinar sua proximidade à solução ótima do problema. Ela é feita através de uma função que melhor representa o problema e tem por objetivo oferecer uma medida de aptidão de cada indivíduo na população corrente, que irá dirigir o processo de busca. Dado um cromossomo, a função de avaliação consiste em se associar um valor numérico de "adaptação", o qual supõe-se proporcional à sua "utilidade" ou "habilidade" do indivíduo representado em solucionar o problema em questão.

3.1.4
Operadores Genéticos

Os operadores mais conhecidos nos algoritmos genéticos são os de Reprodução, Cruzamento (*Crossover*) e Mutação.

Reprodução: refere-se ao processo de selecionar e copiar um determinado cromossomo para a população seguinte de acordo com sua aptidão. Isto significa que os cromossomos mais aptos têm maior probabilidade de contribuir para a formação de um ou mais indivíduos da população seguinte. Existem basicamente os seguintes métodos: troca de toda população, troca de toda população com elitismo, onde todos os cromossomos são substituídos, sendo o cromossomo mais apto da população corrente copiado para população seguinte, e troca parcial da população (*steady state*), onde os M melhores indivíduos da população corrente são copiados para população seguinte (Goldberg, 1989) (Koza, 1992) (Michalevicz, 1996).

Cruzamento: é um operador baseado na troca de partes dos cromossomos (pais), formando-se duas novas soluções (filhos). Este processo pode ser observado no exemplo a seguir (Figura 5), onde a solução está codificada com alfabeto binário.

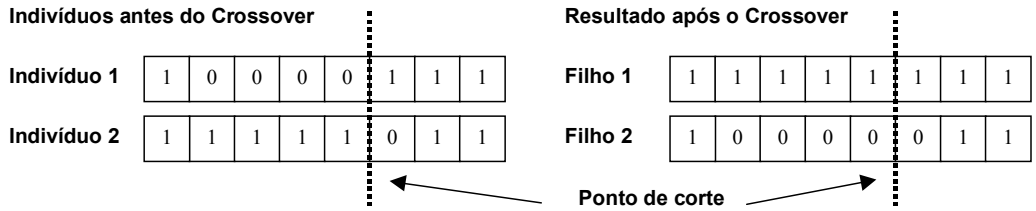

Figura 5 – Cruzamento de um ponto

O ponto onde ocorre o corte para a realização do cruzamento é escolhido aleatoriamente; no exemplo da Figura 3 utilizou-se um único ponto, mas podem ser realizados cortes em mais de um ponto, caracterizando o *multi-point crossover* (Goldberg, 1989) (Michalevicz, 1996) (Holland, 1992). Para realizar o cruzamento, primeiro é necessária a escolha, por sorteio, dos cromossomos "pais". Em seguida ocorre a realização ou não do cruzamento segundo um parâmetro, denominado taxa de cruzamento. Deste modo, de acordo com a taxa de

cruzamento, pode ocorrer que os cromossomos "pais" sejam repassados sem modificação para a geração seguinte, criando "filhos" idênticos a eles.

A idéia do operador de Cruzamento é tirar vantagem (*exploit*) do material genético presente na população.

Mutação: é a troca aleatória do valor contido nos genes de um cromossomo por outro valor válido do alfabeto. No caso de alfabeto binário troca-se de 0 para 1 e vice-versa. Da mesma forma que para o cruzamento, utiliza-se uma taxa de mutação que, para cada bit da seqüência de caracteres, sorteia-se se ocorrerá ou não a mutação; no caso de ocorrência, o bit será trocado por outro valor válido pertencente ao alfabeto (Figura 6).

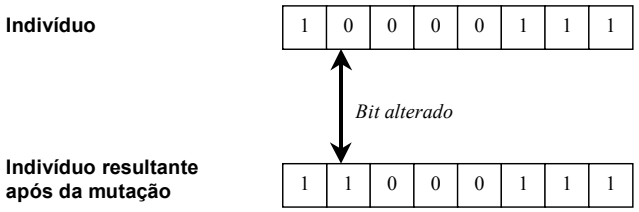

Figura 6 – Mutação

A mutação garante a diversidade das características dos indivíduos da população e permite que sejam introduzidas informações que não estiveram presentes em nenhum dos indivíduos. Além disto, proporciona uma busca aleatória (*exploration*) no AG, oferecendo oportunidade para que mais pontos do espaço de busca sejam avaliados.

3.1.5 Parâmetros da Evolução

Os parâmetros que mais influenciam no desempenho do algoritmo genético são:

Tamanho da População: afeta o desempenho global e a eficiência dos Algoritmos Genéticos. Uma população muito pequena oferece uma pequena cobertura do espaço de busca, causando uma queda no desempenho. Uma população suficientemente grande fornece uma melhor cobertura do domínio do problema e previne a convergência prematura para soluções locais. Entretanto, com uma grande população tornam-se necessários recursos computacionais

maiores, ou um tempo maior de processamento do problema. Logo, deve-se buscar um ponto de equilíbrio no que diz respeito ao tamanho escolhido para a população.

Taxa de Cruzamento: probabilidade de um indivíduo ser recombinado com outro. Quanto maior for esta taxa, mais rapidamente novas estruturas serão introduzidas na população. Entretanto, isto pode gerar um efeito indesejável, pois a maior parte da população será substituída, causando assim perda de variedade genética, podendo ocorrer perda de estruturas de alta aptidão e convergência a uma população com indivíduos extremamente parecidos, indivíduos estes de solução boa ou não. Com um valor baixo, o algoritmo pode-se tornar muito lento para oferecer uma resposta aceitável.

Taxa de Mutação: probabilidade do conteúdo de um gene do cromossomo ser alterado. A taxa de mutação previne que uma dada população fique estagnada em um valor, além de possibilitar que se chegue em qualquer ponto do espaço de busca. Porém, deve-se evitar uma taxa de mutação muito alta, uma vez que esta pode tornar a busca essencialmente aleatória, prejudicando fortemente a convergência para uma solução ótima.

Intervalo de Geração: controla a porcentagem da população que será substituída durante a próxima geração (substituição total, substituição com elitismo, substituição dos piores indivíduos da população atual, substituição parcial da população sem duplicatas). Esse número de indivíduos substituídos também é conhecido como GAP.

Número de gerações: representa o número total de ciclos de evolução de um Algoritmo Genético, sendo este um dos critérios de parada do algoritmo genético. Um número de gerações muito pequeno causa uma queda no desempenho; um valor grande faz necessário um tempo maior de processamento, mas fornece uma melhor cobertura do domínio do problema, evitando a convergência para soluções locais.

3.3
Co-evolução

Para se aplicar algoritmos evolucionários com sucesso em problemas com complexidade cada vez maior, torna-se necessário introduzir noções explícitas de

modularidade nas soluções para que elas disponham de oportunidades razoáveis de evoluir na forma de subcomponentes co-adaptados (Potter & Jong, 2000).

Existem duas razões principais pelas quais algoritmos evolucionários convencionais não são totalmente adequados para resolver este tipo de problema. Em primeiro lugar, os algoritmos genéticos convencionais impedem, a longo prazo, a preservação de certos componentes da solução pois, por estarem codificados por completo em um indivíduo, eles são avaliados como um todo e apenas os subcomponentes que pertencem a indivíduos com avaliações altas serão preservados. Em segundo lugar, o fato da representação estar relacionada a uma solução completa e por não haver interações entre os membros da população, não existe pressão evolucionária para a ocorrência de co-adaptação, ou seja, não existe pressão para a adaptação de um subcomponente dada a ocorrência de uma mudança em outro subcomponente (Potter & Jong, 1994) (Potter & Jong, 2000).

A decomposição do problema no entanto tem um aspecto importante que deve ser levado em conta e que está relacionado com a evolução de subcomponentes interdependentes. Se for possível decompor o problema em subcomponentes independentes, cada um deles pode evoluir sem haver necessidade de se preocupar com os outros. Pode-se visualizar isto imaginando-se que cada subcomponente evolui no seu próprio espaço de busca, desacoplado do espaço de busca dos outros componentes. Entretanto, a maioria dos problemas só pode ser decomposta em subcomponentes que possuem uma interdependência complexa. O efeito de mudar um desses subcomponentes provoca uma "deformação" no espaço de buscas dos outros subcomponentes que estão acoplados por esta interdependência. É possível visualizar isto graficamente da seguinte forma: suponha que se deseje minimizar uma função de duas variáveis e que essas duas variáveis formem dois subcomponentes interdependentes. Esta função pode ser por exemplo descrita pela equação:

$$f(x,y) = x^2 + (x+y)^2 \tag{1}$$

O gráfico desta equação é mostrado a seguir.

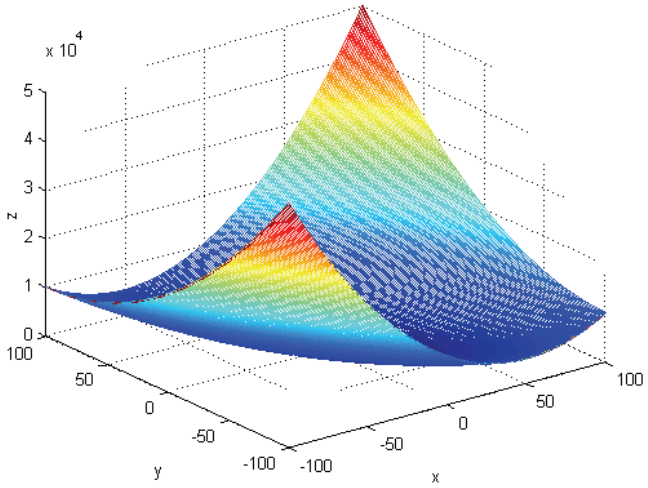

Figura 7 – Gráfico da equação

Supondo que se deseje encontrar o valor mínimo desta função, que ocorre quando *x* e *y* são iguais a zero (sendo então *f(x,y)* igual a zero), o espaço de busca da variável y assume a seguinte forma quando *x* é igual a 40:

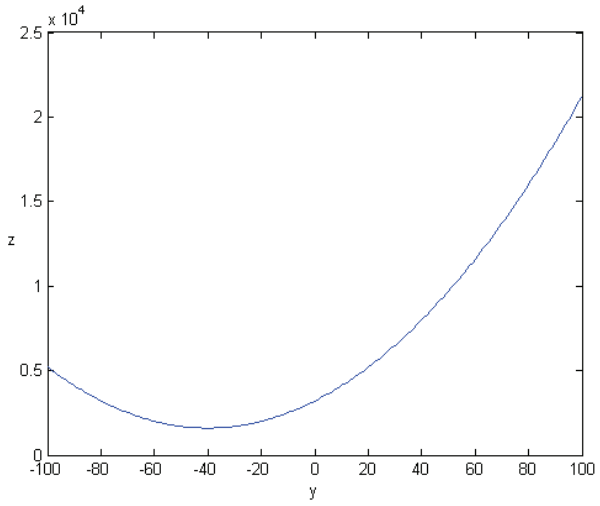

Figura 8 – Gráfico para x igual a 40

Quando a variável *x* por outro lado, assume o valor -40, o espaço de busca da variável *y* se deforma e assume a seguinte forma:

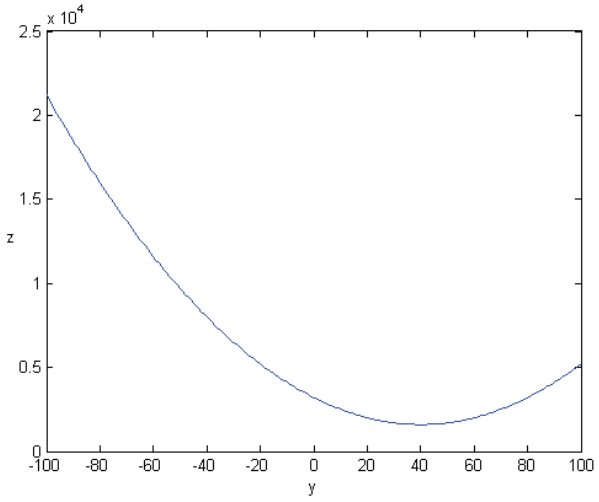

Figura 9 – Gráfico para x igual a -40

Da Figura 8 para a Figura 9 pode-se notar a deformação do espaço de busca da variável *y* quando muda-se o valor da variável *x*. No primeiro caso, o mínimo no espaço de buscas de *y* encontra-se em -40 enquanto que no segundo caso o mínimo encontra-se em 40.

Espera-se, portanto, que com o desacoplamento e com a capacidade de adaptação dos algoritmos evolucionários os mesmos consigam se adaptar facilmente a este tipo de dinâmica do problema.

Várias pesquisas foram feitas visando estender o modelo evolucionário básico de modo que o mesmo suportasse subcomponentes co-adaptados. Uma das primeiras extensões feitas foi o modelo de sistemas classificadores (Holland, 1978) (Holland, 1986). Em Giordana et al. (1994) tem-se um sistema (chamado REGAL) que aprende regras de classificação e que utiliza um operador chamado "voto universal" (*universal suffrage*) para fazer a decomposição do problema. Em Giordana & Neri (1996) o desempenho do sistema REGAL foi melhorado através do uso de ilhas populacionais semi-isoladas. Modelos com múltiplas espécies, ou seja, que incorporam populações geneticamente isoladas também têm sido usados para evoluir subcomponentes co-adaptados. Trabalhos nesta área incluem modelos hospedeiros-parasitas (*host-parasite*) para redes de ordenação onde uma espécie (os hospedeiros) evolui redes de ordenação para serem aplicadas em casos de teste representados pela outra espécie (os parasitas) como uma lista de números a serem ordenados (Hillis, 1991). Outros modelos com duas espécies competindo foram

usados para solucionar problemas de aprendizado de jogos, fazendo com que cada uma das espécies representasse um dos jogadores (Rosin & Belew, 1995).

Alguns outros pesquisadores exploraram o uso de modelos com espécies em cooperação como, por exemplo, para a função de três bits de Goldberg (Paredis, 1995) e para a otimização de funções (Potter & Jong, 1994). Finalmente, em Potter & Jong (2000), procurou-se definir um modelo co-evolucionário cooperativo genérico.

Nesta arquitetura duas ou mais espécies diferentes formam um ecossistema. Como na natureza, as espécies são geneticamente isoladas ou seja, os indivíduos só podem se reproduzir com outros indivíduos da mesma espécie. Isto é feito simplesmente isolando-se as espécies em populações separadas. As espécies somente interagem umas com as outras através de um domínio compartilhado e têm uma relação apenas de cooperação.

O modelo co-evolucionário cooperativo genérico (Potter & Jong, 2000) é mostrado na Figura 10.

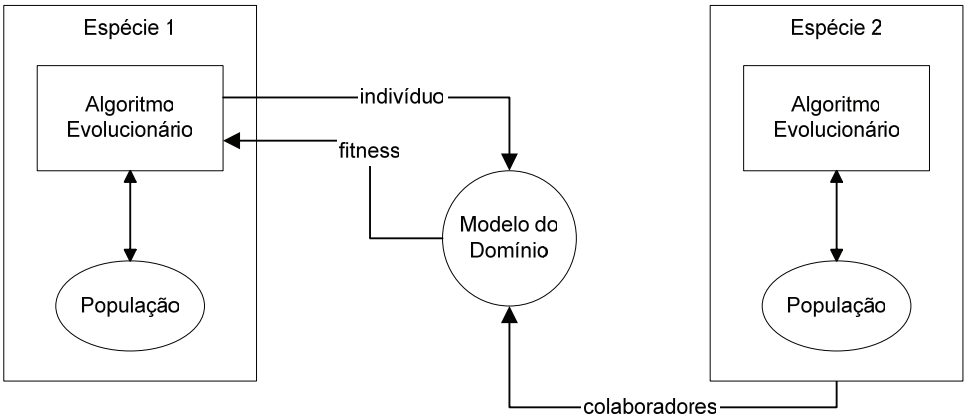

Figura 10 – Modelo co-evolucionário genérico

Apesar de serem mostradas apenas duas espécies neste modelo, o mesmo pode ser usado para *n* espécies diferentes. Cada espécie evolui em sua própria população (pois como já foi mencionado, elas são isoladas geneticamente) e se adaptam ao ambiente através de repetidas aplicações do algoritmo evolucionário. Para se calcular a aptidão dos indivíduos de uma determinada espécie, deve-se submetê-lo ao modelo do domínio (que contém a função de avaliação) juntamente

com um ou mais colaboradores de cada uma das outras espécies (de modo a formar a solução completa).

3.4
Estratégias de avaliação multi-objetivo

Embora a maioria dos problemas do mundo real requerem a otimização simultânea de múltiplos, freqüentemente competitivos, critérios (ou objetivos), a solução para tais problemas é usualmente computada através da combinação dos objetivos em um único critério a ser otimizado, de acordo com alguma função. Em vários casos, porém, a melhor forma de efetuar a combinação de objetivos não é conhecida antes do processo de otimização. Então o problema deve ser tratado como um Problema de Otimização Multi-Objetivo com objetivos não comensuráveis.

Um problema de otimização Multi-Objetivo trabalha com mais de uma função objetivo. Técnicas tradicionais de otimização têm sido usadas para solucionar estes problemas no passado. Estas técnicas originalmente foram formuladas para trabalhar com uma única função objetivo e encontrar uma solução ótima. Mas novas técnicas para considerar os objetivos separadamente estão sendo aplicadas. Assim, os vários objetivos não são combinados em uma única função. Um resumo dessas técnicas será mostrado nesta seção (Deb, 2001).

3.4.1
Formulação

Um problema de otimização multi-objetivo possui um número de funções objetivo a serem otimizadas (maximizar ou minimizar). Além disso, possui restrições que devem ser satisfeitas por qualquer solução factível. O enunciado otimização multi-objetivo é o seguinte:

$$\left.\begin{array}{rll} \text{maximizar/minimizar} & f_m(x), & m = 1,2,...,M \\ \text{restrita a} & g_j(x) \geq 0, & j = 1,2,...,J; \\ & h_k(x) = 0, & k = 1,2,...,K; \\ & x_i^{(L)} \leq x_i \leq x_i^{(U)}, & i = 1,2,...,n. \end{array}\right\} \quad (2)$$

onde x é o vetor de n variáveis de decisão $x = (x_1, x_2, ..., x_n)^T$. Os valores $x_i^{(L)}$ e $x_i^{(U)}$, representam o mínimo e o máximo valor respectivamente para a variável x_i. Estes limites definem o *espaço de variáveis de decisão* ou *espaço de decisão D*. O vetor *x* será referido também como *solução*.

As *J* desigualdades (g_j) e as *K* igualdades (h_k) são chamadas de funções de restrição. Uma solução *x* factível será aquela que satisfaça as *J* + *K* funções de restrição e os 2*n* limites. Caso contrário a solução será não factível. O conjunto de todas as soluções factíveis formam a *região factível* ou *espaço de busca S*.

Cada uma das *M* funções objetivo $f_1(x), f_2(x), ..., f_M(x)$ pode ser maximizada ou minimizada. Porém, para trabalhar com os algoritmos de otimização, é necessário converter todas para serem maximizadas ou minimizadas. O vetor funções objetivo **f**(x) conforma um espaço multi-dimensional chamado *espaço de objetivos Z*. Esta é uma diferença fundamental em relação à otimização de objetivos simples, cujo espaço de objetivos é uni-dimensional. O mapeamento acontece, então, entre um vetor *x* (*n*-dimencional) e um vetor **f**(x) (*M*-dimensional). Por exemplo, se cada elemento de *x* e **f**(x) são números reais, então **f**(x) estaria mapeada como $\mathbf{f}(x) : \Re^n \rightarrow \Re^M$.

3.4.2
Soluções Pareto-ótimas

A tomada de decisões implica num processo que consiste em vários fatores, com o objetivo de encontrar a melhor solução. Em alguns casos, podem aparecer várias soluções boas, das quais nenhuma é quantitativamente melhor que a outra. Por exemplo, na hora de comprar um carro suponha-se que se está procurando o preço e conforto. A Figura 11 ilustra várias alternativas de escolha.

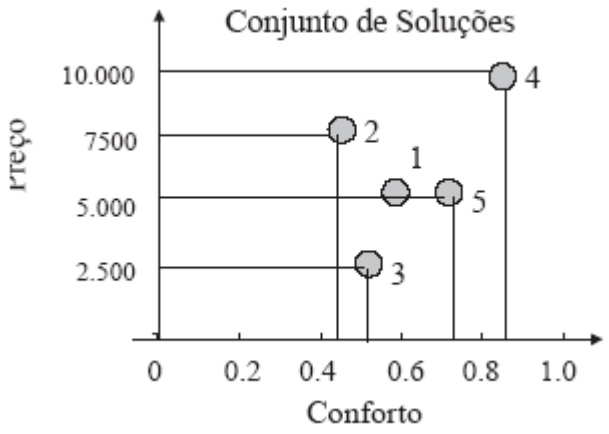

Figura 11 – Exemplo que ilustra várias opções de compra de carro (1-5), considerando o seu custo e conforto

O objetivo é minimizar o custo e maximizar o conforto. Neste caso tem-se cinco possíveis opções de compra. Intuitivamente, descarta-se a solução 1, já que a solução 5 fornece mais conforto por igual preço. A solução 2 é descartada pela mesma razão. Tem-se então três soluções: 3,4,5, que são boas alternativas de compra. Em termos quantitativos nenhuma é melhor que a outra, pois uma é mais confortável mas menos barata, e vice-versa. Existe então um *compromisso* entre os objetivos. Quanto maior o conforto, maior o preço caro e vice-versa.

Diz-se que uma solução domina uma outra se seus valores são melhores em todos os objetivos. Por exemplo, a solução 5 domina a solução 1. Já a solução 5 é não dominada por nenhuma outra. O mesmo acontece com as soluções 3 e 4. Se não se conhece a priori a importância relativa de cada objetivo, pode-se dizer que as soluções 3,4, e 5 são igualmente boas. Portanto, existe um conjunto de soluções ótimas; este conjunto é chamado de conjunto não dominado. As outras soluções (1, e 2) formam o conjunto dominado. Estes conjuntos têm as seguintes propriedades:

1. Qualquer par de soluções do conjunto não dominado deve ser não-dominado um em relação ao outro.

2. Quaisquer das soluções não pertencentes ao conjunto não-dominado, devem ser dominadas por no mínimo uma solução no conjunto não-dominado.

Se os pontos não dominados estão em um espaço contínuo, pode-se desenhar uma curva. Todos os pontos contidos na curva formam a *Frente de Pareto* ou *Fronteira de Pareto*.

Quando a informação adicional sobre importância dos objetivos é desconhecida, todas as soluções Pareto-ótimas são igualmente importantes. Deb assinala duas importantes metas em Otimização Multi-Objetivo (Deb, 2001):

1. Encontrar um conjunto de soluções o mais próximo possível da *Fronteira de Pareto*.

2. Encontrar um conjunto de soluções com a maior diversidade possível.

A primeira meta é comum para qualquer processo de otimização. Soluções muito distantes da *Fronteira de Pareto* não são desejáveis. Porém, encontrar a maior diversidade dentro das soluções é uma meta específica para Otimização Multi-Objetivo. Na Figura 12a mostra-se uma boa distribuição de soluções na fronteira de Pareto, enquanto que na Figura 12b as soluções estão distribuídas apenas em algumas regiões. É necessário assegurar a maior cobertura possível da fronteira, já que isso implica em ter um bom conjunto de soluções *comprometidas* com os objetivos desejados. Como em Problemas de Otimização Multi-Objetivo trabalha-se com o espaço de decisões e o espaço de objetivos, é desejável que as soluções tenham uma boa diversidade nestes espaços. Normalmente, uma boa diversidade em um destes espaços garante também a diversidade no outro. Entretanto, em alguns problemas isto não acontece.

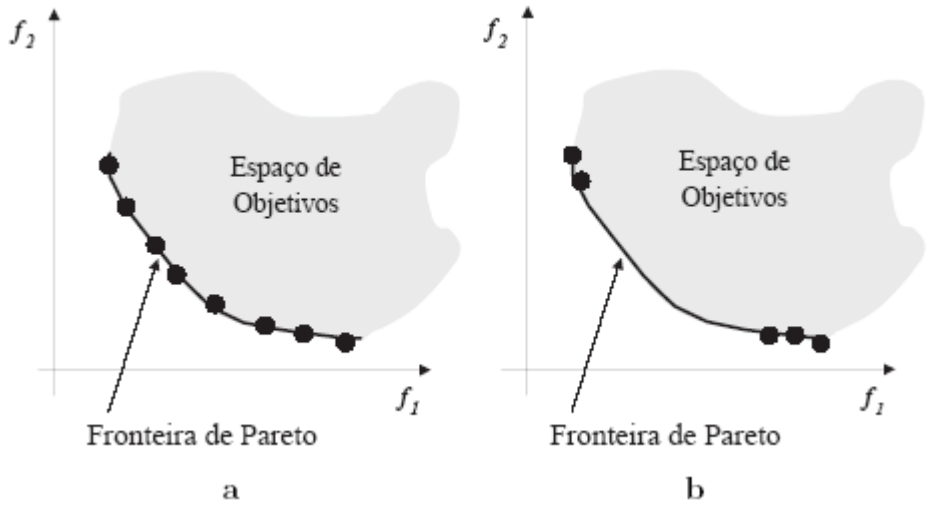

Figura 12 – Distribuição de Soluções na *Fronteira de Pareto*

Geralmente, é necessário um critério adicional para diferenciar as soluções encontradas no conjunto Pareto-Ótimo, de modo que se tenha uma única solução ótima. Isso pode ser feito por algum especialista, após o processo de otimização ter descoberto um conjunto de soluções não-dominadas, ou pode-se adotar algum critério que diferencie as soluções do conjunto Pareto-Ótimo durante o processo de otimização.

No próximo capítulo será descrito o modelo de otimização desenvolvido nesta pesquisa e as técnicas utilizadas para tratar os objetivos separadamente.