3
Métodos de Otimização

3.1.
Introdução

Os problemas de otimização são problemas de maximização ou minimização de função de uma ou mais variáveis num determinado domínio, sendo que, geralmente, existe um conjunto de restrições nas variáveis.

Os algoritmos usados para a solução de um problema de otimização podem ser, basicamente, determinísticos ou probabilísticos.

Neste capítulo são apresentadas as principais características desses métodos, apresentando suas vantagens e desvantagens. Serão abordados de uma maneira mais detalhada os algoritmos de computação evolucionária, que pertencem a uma família de métodos probabilísticos de otimização, visto que este trabalho se baseou em um destes métodos, conhecido como Estratégia Evolutiva (*EE*).

Alguns trabalhos utilizando algoritmos de computação evolucionária vêm sendo desenvolvidos no Departamento de Engenharia Civil da PUC-Rio, dos quais pode-se citar DEL SAVIO (2005), RAMIRES (2004) e BORGES(2003).

3.2.
Definições

Para melhor entendimento dos algoritmos de otimização, faz-se necessário o conhecimento de alguns conceitos e definições utilizados na literatura (BASTOS, 2004). A seguir são listados alguns termos usualmente relacionados a um problema de otimização qualquer:

- Variáveis de projeto: São aquelas que se alteram durante o processo de otimização, podendo ser contínuas (reais), inteiras ou discretas.

- Restrições: São funções de igualdade ou desigualdade sobre as variáveis de projeto que descrevem situações de projeto consideradas não desejáveis.

- Espaço de busca: É o conjunto, espaço ou região que compreende as soluções possíveis ou viáveis sobre as variáveis do projeto do problema a ser otimizado, sendo delimitado pelas funções de restrição.

- Função Objetivo: É a função de uma ou mais variáveis de projeto que se quer otimizar, minimizando-a ou maximizando-a.

- Ponto Ótimo: É o ponto formado pelas variáveis de projeto que extremizam a função objetivo e satisfazem as restrições.

- Valor Ótimo: É o valor da função objetivo no ponto ótimo.

3.3. Métodos Determinísticos

Os métodos de otimização baseados nos algoritmos determinísticos – maioria dos métodos clássicos – geram uma seqüência determinística de possíveis soluções requerendo, na maioria das vezes, o uso de pelo menos a primeira derivada da função objetivo em relação às variáveis de projeto.

Nestes métodos, a função objetivo e as restrições são dadas como funções matemáticas e relações funcionais. Além disso, a função objetivo deve ser contínua e diferenciável no espaço de busca (BASTOS, 2004). Esse tipo de problema pode ser representado matematicamente da seguinte forma:

Maximizar / Minimizar: $f(x_1, x_2, ..., x_n)$

Satisfazendo:

$g_1(x_1, x_2, ..., x_n)\{\leq = \geq\} b_1$

$\vdots$

$g_m(x_1, x_2, ..., x_n)\{\leq = \geq\} b_m$

em que:

$x_1, x_2, ..., x_n$ - variáveis de projeto

$f(x_1, x_2, ..., x_n)$ - função objetivo

$g_1, g_2, ..., g_m$ - restrições

Figura 3.1– Formulação de um problema de otimização.

Quando se trata de um problema de variáveis discretas, considera-se um espaço de busca com variáveis contínuas que, após a otimização, fornecerão uma aproximação das variáveis de projeto para as disponíveis no espaço discreto. Entretanto, isso gera um trabalho adicional na escolha das variáveis discretas mais próximas das contínuas encontradas. Sempre existirão duas opções de variáveis discretas para cada variável contínua, ou seja, uma imediatamente superior e outra imediatamente inferior.

Os métodos determinísticos apresentam teoremas que lhes garantem a convergência para uma solução ótima que não é necessariamente a solução ótima global. Como nesses métodos a solução encontrada é extremamente dependente do ponto de partida fornecido, pode-se convergir para um ótimo local, por isso não possuem bom desempenho em otimizar funções multimodais, isto é, funções que possuem vários ótimos locais.

De acordo com OLIVIERI (2004), BASTOS (2004) e HAFTKA(1993), os problemas de otimização abordados pelos métodos clássicos podem ser classificados em duas classes, conforme as características da função objetivo e das restrições:

- Programação Linear: quando a função objetivo e as restrições são funções lineares das variáveis de projeto. O Método Simplex (HADLEY, 1982) é o método mais tradicional para solucionar este tipo de problema de otimização;

- Programação Não-Linear: quando a função objetivo, ou pelo menos uma das restrições, é uma função não-linear das variáveis de projeto. Nesta classe, os métodos que mais se destacam são: Método de Programação Linear Seqüencial, Método de Programação Quadrática Seqüencial, Método das Direções Viáveis e Método do Gradiente Reduzido, entre outros.

3.4. Métodos Probabilísticos

Os métodos de otimização baseados nos algoritmos probabilísticos usam somente a avaliação da função objetivo e introduzem no processo de otimização dados e parâmetros estocásticos. Por não utilizarem a derivada da função objetivo, são considerados métodos de ordem zero.

São listadas a seguir algumas vantagens dos algoritmos probabilísticos em relação aos algoritmos determinísticos (BASTOS, 2004):

- a função objetivo e as restrições não precisam necessariamente ter uma representação matemática;
- não requerem que a função objetivo seja contínua ou diferenciável;
- trabalham adequadamente, tanto com parâmetros contínuos quanto com discretos, ou ainda com uma combinação deles;
- não necessitam de formulações complexas ou reformulações para o problema;
- não há restrição alguma quanto ao ponto de partida dentro do espaço de busca da solução;
- realizam buscas simultâneas no espaço de possíveis soluções através de uma população de indivíduos;
- Otimizam um grande número de variáveis, desde que a avaliação da função objetivo não tenha um custo computacional demasiadamente alto.

A maior desvantagem em relação aos métodos clássicos é o tempo de processamento.

3.4.1. Computação Evolucionária

Segundo BÄCK et al.(1997), a Computação Evolucionária teve origem no final da década de 50 e permaneceu relativamente desconhecida da comunidade

científica por aproximadamente três décadas, devido principalmente à falta de computadores eficientes na época, mas também devido à metodologia pouco desenvolvida durante as primeiras pesquisas. Durante a década de setenta, os trabalhos de Holland, Rechenberg, Shwefel e Foger foram fundamentais para modificar a imagem da Computação Evolucionária que, a partir de então, começou a ser largamente desenvolvida.

Os Algoritmos Evolucionários ($AE's$) formam uma classe de métodos de otimização probabilísticos que são inspirados por alguns princípios baseados em mecanismos evolutivos encontrados na natureza, como auto-organização e o comportamento adaptativo (BEYER et al, 2002).

De acordo com BARBOSA (1997), um algoritmo evolucionário se distingue dos métodos determinísticos mais comuns basicamente por:

- empregar uma população de indivíduos, ou soluções;
- trabalhar sobre uma codificação das possíveis soluções (genótipos) e não sobre as soluções (fenótipos) propriamente ditas;
- empregar regras de transição probabilísticas;
- não requerer informações adicionais (derivadas, por exemplo) sobre a função a otimizar e as restrições.

Assim, a busca de soluções pode se dar em conjuntos não-convexos com funções objetivo também não-convexas e não-diferenciáveis podendo-se trabalhar simultaneamente com variáveis reais, lógicas e inteiras. Vale ressaltar também que os $AE's$ não são facilmente presos a mínimos locais como é o caso dos algoritmos usuais dos métodos determinísticos. Ao utilizar um AE, essas características podem levar à descoberta de soluções não convencionais que não poderiam ser vislumbradas por serem contra-intuitivas. É um paradigma que não exige conhecimento prévio de uma maneira de encontrar a solução.

Para a utilização de AE em problemas de otimização com restrições, uma das possibilidade é utilizar um método de penalização. Isso pode ser feito através da pena de morte, onde um indivíduo é simplesmente eliminado da população quando violar as restrições ou quando não for possível avaliar sua *aptidão*[*]. Porém, possui a desvantagem de poder estar descartando um indivíduo potencialmente útil ao processo evolutivo. Outra maneira seria introduzir uma

[*] Utilizou-se a palavra "*aptidão*" como tradução da palavra "*fitness*" usualmente adotada na literatura inglesa para se referir ao desempenho de um indivíduo da população.

função de penalização para incorporar as restrições à função objetivo, de maneira análoga ao que se faz nos métodos clássicos de otimização, reduzindo a *aptidão* dos indivíduos que violam as restrições (BARBOSA, 1997).

Para ilustrar o comportamento de um AE, considera-se uma função objetivo unidimensional a ser maximizada. A Figura 3.2 mostra três etapas da busca evolucionária, mostrando como os indivíduos são distribuídos no começo (a), meio (b) e fim (c) do processo de evolução. Na primeira fase, imediatamente após a inicialização da população, os indivíduos são aleatoriamente espalhados em todo o espaço de busca. Depois de algumas gerações a distribuição modifica-se: devido aos operadores de variação e seleção, a população abandona as regiões de baixa *aptidão* e começa a ocupar áreas de maior *aptidão*. No final da busca, tendo sido escolhida uma condição de parada apropriada, toda a população está concentrada em torno de poucos pontos, onde alguns desses pontos podem ser sub-ótimos. Pode ocorrer de todos os membros da população se posicionarem em torno de um ótimo local ao invés de um ótimo global. Essa convergência prematura é um efeito conhecido de perda rápida de diversidade, que leva a população a ficar presa a ótimos locais (EIBEN & SMITH, 2003).

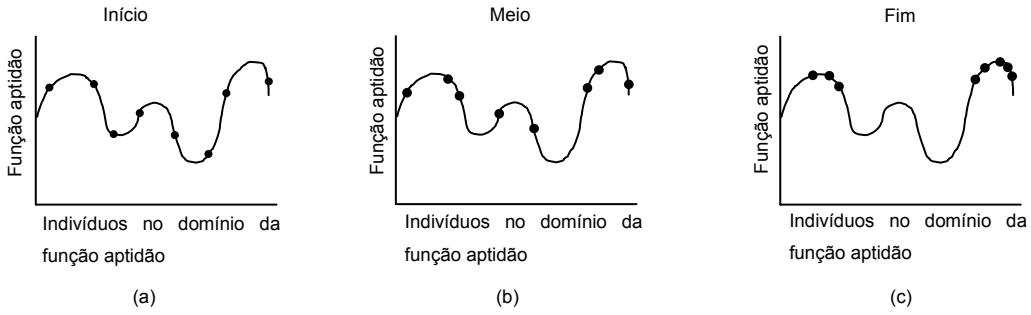

Figura 3.2 – Evolução típica de um AE, ilustrada de acordo com a distribuição da população. Adaptado de EIBEN & SMITH (2003).

Conforme CORTES & SAAVEDRA (2000), a Computação Evolucionária tem sido utilizada com sucesso para resolução de complexos problemas de otimização. Seu principal obstáculo é a precisão da solução a ser encontrada, pois o quanto mais próximo da solução ótima se deseja chegar, mais poder computacional e tempo de processamento são exigidos, principalmente quando são utilizadas funções multimodais.

3.4.1.1. Definições

Para a utilização de um AE são necessárias algumas definições adicionais que são particulares a esse tipo de algoritmo (BASTOS, 2004; EIBEN & SMITH, 2003). Como a Computação Evolucionária é baseada em mecanismos evolutivos encontrados na natureza, muitos termos adotados pelos $AE's$ baseiam-se na Genética, tais como:

- Cromossomo ou genótipo– representa um indivíduo no espaço do AE, ou seja, representa um indivíduo codificado;
- Fenótipo – representa um indivíduo no espaço de busca original;
- Indivíduo – é um membro da população;
- Gene – unidade básica do cromossomo, ou seja, é um elemento do vetor que representa o cromossomo;
- População – conjunto de indivíduos ou cromossomos;
- Geração – ordem evolutiva das diferentes populações;
- Operações genéticas – conjunto de operações que o AE realiza sobre cada um dos cromossomos;
- Função aptidão – quando o AE é utilizado em um problema de otimização, a função aptidão equivale à função objetivo.

3.4.1.2. Algoritmo Evolucionário

A principal idéia em que se baseia qualquer variação de um Algoritmo Evolucionário é: dada uma população de indivíduos, a pressão do meio ambiente causa uma seleção natural que evolui a população. Sendo assim, qualquer algoritmo evolucionário deve ter as seguintes componentes básicas para resolver um problema (MICHALEWICZ, 1996; EIBEN & SMITH, 2003; BARBOSA, 1997; BÄCK et al, 1997):

- Uma representação genética das soluções do problema;

  A representação ou codificação de um indivíduo quando se utiliza um AE consiste em relacionar o espaço real do problema com o espaço adotado pelo AE, ou seja, representar/codificar os elementos do espaço real no espaço do AE. Cada elemento do espaço de busca é denominado fenótipo e sua representação no espaço do AE é denominado genótipo.

  Para ilustrar esse processo, considere que em um problema de otimização bidimensional de números inteiros que adote um AE com representação binária, onde o alfabeto é composto dos símbolos 0 e 1, $x = \{x_1, x_2\}$ seja uma possível solução do problema. Sendo o cromossomo codificado com cinco *bits* para cada uma das variáveis do problema, elas podem ser representadas da seguinte maneira:

  x_1 =00100

  x_2 =10100

  Essas codificações seriam os genes que concatenados formam o cromossomo, que representa uma possível solução do problema:

  0010010100

  Para recuperar os valores das variáveis no espaço real, ou seja, obter o fenótipo, é necessário um processo de descodificação:

  IND_1 = 0x2^4 + 0x2^3 + 1x2^2 + 0x2^1 + 0x2^0 = 4

  IND_2 = 1x2^4 + 0x2^3 + 1x2^2 + 0x2^1 + 0x2^0 = 20

  Para um problema com variáveis inteiras, o valor da variável é igual ao próprio índice fornecido pela codificação (IND). No caso de variáveis discretas, a decodificação fornece um índice que localiza o valor da variável numa lista de referência, que representa o espaço de busca para esta variável (BASTOS, 2004).

  Para as variáveis contínuas, tem-se a seguinte decodificação:

$$x_i = x_i^L + IND_i \frac{x_i^U - x_i^L}{2^{nb} - 1} \tag{3.1}$$

Onde:

x - ponto de busca no espaço.

x^L - limite inferior do espaço de busca;

x^U - limite superior do espaço de busca;

nb - número de bits;

IND -Índice fornecido pela decodificação da variável;

i - número de variáveis;

Segundo BASTOS (2004), a utilização de codificação binária é dada pelas seguintes razões:

- Extrema facilidade para criar e manipular vetores binários;
- Utiliza rigorosamente a precisão determinada para cada variável;
- Altamente indicada para se operar com variáveis discretas.

Porém, quando o problema em análise necessita que as variáveis envolvidas sejam de alta precisão numérica, a codificação binária possui enorme desvantagem pois, neste caso, faz-se necessário que os cromossomos possuam um comprimento extremamente grande, reduzindo a performance do AE. Outra desvantagem é a necessidade constante de conversão entre os valores reais e os binários nas diversas iterações do processo.

- População

  O papel da população é manter as possíveis soluções. Enquanto os indivíduos são estáticos, isto é, não se modificam, a população é uma unidade de evolução. Dada uma representação, definir uma população equivale a decidir o número de indivíduos que irão formá-la. Em alguns $AE's$ mais sofisticados a população pode ter uma estrutura adicional, com medidas de distância ou relações de vizinhança. Em quase todas as aplicações de AE o tamanho da população é constante, não sendo modificado durante a evolução.

- Uma maneira de inicializar a população;

  A inicialização da população geralmente é simples na maioria das aplicações de AE, e é feita gerando indivíduos aleatoriamente. Porém, algumas heurísticas podem ser usadas para gerar uma população inicial

com maior *aptidão,* como, por exemplo, iniciar a população com soluções aproximadas conhecidas ou contendo algum tipo de informação prévia. Se isso vale o esforço computacional extra envolvido, depende muito da aplicação.

- Uma função aptidão

  A função aptidão é a responsável pelo processo de seleção dos indivíduos e deve indicar a qualidade de cada indivíduo na população, sendo assim, influi diretamente na evolução da população. Tecnicamente, é uma função que designa uma medida de qualidade ao genótipo, ou seja, a *aptidão*.

- Operadores genéticos

  Os operadores genéticos alteram a composição genética dos filhos durante a reprodução. O papel dos operadores é criar novos indivíduos a partir dos antigos. Os operadores trabalham sobre a codificação das possíveis soluções (genótipo) e não sobre as soluções (fenótipos) propriamente ditas. Os principais operadores são recombinação e mutação.

  A recombinação é um operador que une informações de dois ou mais genótipos pais para gerar um ou dois descendentes. O operador de recombinação é estocástico, isto é, é aleatória a escolha de que partes de cada pai será recombinada e o modo que estas partes serão recombinadas.

  A mutação é um operador que após ser aplicado a um genótipo gera um filho. Similar a recombinação, a mutação é um operador sempre estocástico: seu resultado – o filho – depende dos resultados de uma série de escolhas aleatórias.

- Um mecanismo de seleção

  O papel da seleção é diferenciar os indivíduos baseados nas suas qualidades, em particular, permitir que os melhores indivíduos tornem-se pais da próxima geração.

- Um critério de parada

  Caso o problema tenha um valor ótimo da função aptidão conhecido, o critério de parada pode ser quando este valor for atingido, considerando uma certa precisão. Porém, como $AE's$ são estocásticos e não há garantias de que o valor ótimo será atingido, essa condição pode nunca

ser satisfeita e o algoritmo nunca parar. As opções comumente usadas como critério de parada são:

1. tempo máximo transcorrido;
2. o número total de avaliações da função aptidão atingir um número limite;
3. quando a *aptidão* melhorar muito pouco durante um certo período de tempo (ou um certo número de gerações ou um certo número de avaliações da função aptidão);
4. quando a diversidade da população diminuir até um certo limite, sendo diversidade uma medida do número de diferentes soluções presente na população, que pode ser medido pelas diferentes *aptidões* presentes na população ou pelo número de diferentes fenótipos ou genótipos presentes.

A partir do que foi visto acima, percebe-se que a combinação da aplicação de variação, através dos operadores genéticos, e seleção levam a melhorar o valor da *aptidão* e, em conseqüência, melhorar a população. Pode-se perceber essa evolução como se fosse um processo de otimização, através da busca de valores ótimos, que, no decorrer do processo, ficam cada vez mais próximos. Alternativamente, essa evolução é vista como um processo de adaptação. Deste ponto de vista, a *aptidão* não é vista como uma função objetivo a ser otimizada, mas como uma necessidade do meio ambiente. O processo evolutivo faz a população adaptar-se ao meio ambiente cada vez melhor. A seguir é mostrado um pseudo-código que representa um algoritmo evolucionário.

Geração = 0
Inicializa população (P) ;
Avalia os indivíduos;
Enquanto o critério de parada não for satisfeito repita:

1. Recombinação
2. Mutação
3. Avaliação dos descendentes
4. Seleção
5. Geração = Geração +1

Figura 3.3 – Esquema geral de um Algoritmo Evolucionário. Adaptado de BÄCK et al (1997).

Porém, para que a implementação de um algoritmo evolucionário tenha sucesso quando aplicado a um problema real, as componentes listadas acima requerem algumas heurísticas adicionais, que estão relacionadas à representação genética das soluções, aos operadores que alteram suas composições, aos valores de vários parâmetros, aos métodos de inicialização da população e até mesmo à própria função aptidão.

3.4.1.3. Principais Ramos da Computação Evolucionária

A Computação Evolucionária é uma das áreas da Inteligência Artificial, juntamente com as Redes Neurais e os Sistemas de Lógica Nebulosa (Figura 3.4). A maioria das implementações de algoritmos evolucionários vem de três ramos fortemente relacionados, porém independentemente desenvolvidos (BEYER, 2002 e BÄCK et al, 1997):

- Algoritmos Genéticos ($AG's$);
- Programação Evolutiva ($PE's$);
- Estratégias Evolutivas ($EE's$).

Além dos ramos citados acima, alguns autores, com MICHALEWICZ (1996), citam ainda a Programação Genética ($PG's$) como um importante ramo da Computação Evolucionária.

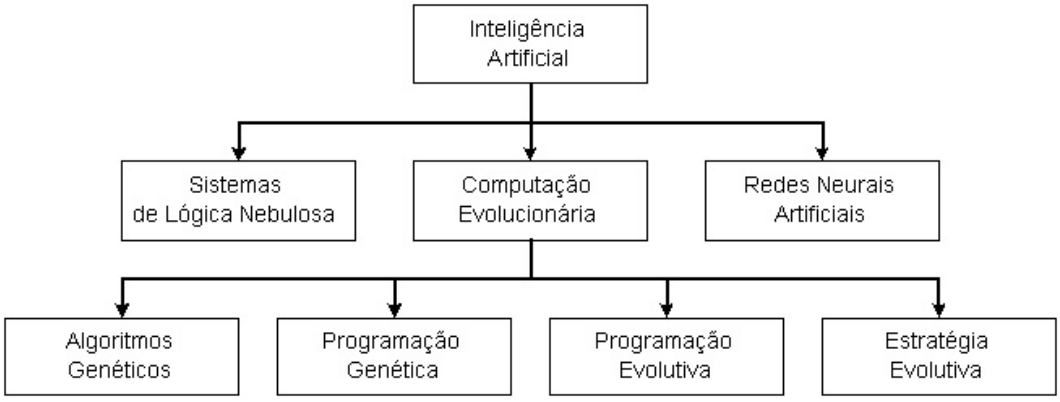

Figura 3.4 – Ramificação da Inteligência Artificial. Adaptada de OLIVIERI (2004).

As principais diferenças entre esses ramos estão na representação dos indivíduos, nos operadores utilizados (mutação e/ou recombinação) e no mecanismo de seleção, embora ultimamente a fronteira entre eles vem se tornando menos nítida.

3.4.1.4. Algoritmos Genéticos *(AG´s)*

Segundo BARBOSA (1977) e BÄCK (1997), o AG foi desenvolvido principalmente por John Holland no final da década de 60 buscando inspiração no que se conhece sobre o processo de evolução natural, conhecimento este iniciado solidamente com a teoria da evolução de Darwin no seu famoso livro *A Origem das Espécies*.

Na maioria das aplicações que utilizam $AG's$, a forma mais comum de construção de uma codificação é utilizar uma cadeia binária, de comprimento fixo. Isso ocorre porque a teoria dos $AG's$ foi desenvolvida com base nesta representação, mas DAVIS (1991) acha que essa representação não é natural e é desnecessária na maioria dos casos.

O principal operador é a recombinação, também conhecido como *crossover* na literatura inglesa, e a mutação é vista como um operador de pequena importância. De forma simplificada, no alfabeto binário, os operadores funcionam da seguinte maneira:

- A mutação é definida pela modificação do símbolo ocorrente em uma posição do cromossomo: se 1 ele passa a 0 e vice-versa. A probabilidade p_m de ocorrência de mutação de um gene é geralmente muito pequena, da ordem de $1/\ell$, onde ℓ é número de bits do cromossomo.

- O *crossover*, no algoritmo padrão, é chamado *crossover de um ponto*. Através de um esquema de seleção implementado, dois indivíduos são escolhidos e, com probabilidade p_c, são submetidos à operação de recombinação. Uma posição de *crossover* é sorteada e o material genético dos pais é recombinado conforme o esquema abaixo:

  p_1 : 1111111 $\qquad\qquad$ f_1 : 1111000

  p_2: 0000000 $\qquad\qquad$ f_2 : 0000111

  Existem outras variações deste operador que podem ser empregadas, como *crossover de dois pontos*, *crossover uniforme*, etc.

- A seleção é tipicamente implementada utilizando um esquema probabilístico. A probabilidade p_i de seleção do i-ésimo indivíduo da população vir a ser selecionado é proporcional à sua aptidão relativa, conforme equação 3.2 (BÄCK et al, 1997;BARBOSA,1977).

$$p_i = \frac{fi}{\sum_{i=1}^{m} fi} \tag{3.2}$$

Onde $fi = f(x_i)$ é assumida positiva e *m* é o número de indivíduos da população.

Um método que aplica essa técnica é o Método da Roleta (*roullete wheel selection*, na literatura inglesa), onde indivíduos de uma geração são escolhidos para fazer parte da próxima geração, através de um sorteio de roleta. Os indivíduos são representados na roleta proporcionalmente ao seu índice de aptidão. Finalmente, a roleta é girada um determinado número de vezes, dependendo do tamanho da população, e são escolhidos como indivíduos que participarão da próxima geração, aqueles sorteados na roleta (Figura 3.5).

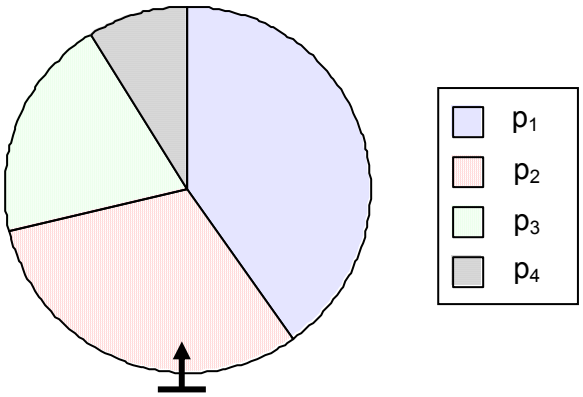

Figura 3.5– Seleção utilizando o método da roleta (Barbosa, 1977).

3.4.1.5. Programação Genética *(PG)*

O paradigma da PG foi desenvolvido por John Koza (KOZA,1992). Segundo MICHALEWICZ (1996), esta técnica constitui uma maneira de fazer uma busca no espaço de possíveis programas computacionais para escolher o melhor deles, ou seja, é uma técnica de geração automática de programas de computador, onde a partir de especificações de comportamento, o computador deve ser capaz de induzir um programa que as satisfaça (KOZA, 1992).

Conforme descrito por RODRIGUES (1992), a técnica baseia-se na combinação de idéias da teoria da evolução (seleção natural), genética (reprodução, cruzamento e mutação), inteligência artificial (busca heurística) e teoria de compiladores (representação de programas como árvores sintáticas).

Os programas são formados pela livre combinação de funções e terminais adequados ao domínio do problema. Parte-se de dois conjuntos: F como sendo o conjunto de funções e T como o conjunto de terminais. O conjunto F pode conter operadores aritméticos (+, -, * etc), funções matemáticas (seno, logaritmo etc), operadores genéticos (E, OU etc) dentre outros. Cada $f \in F$ tem associada uma aridade (número de argumentos) superior a zero. O conjunto T é composto pelas variáveis, constantes e funções de aridade zero (sem argumentos).

O processo evolutivo ocorre a partir da aplicação dos operados genéticos a população e pelo processo de seleção, que é baseado na *aptidão* dos programas, até atingir um determinado critério de parada.

Usualmente, para avaliar a *aptidão* é fornecido um conjunto de casos de treinamento, contendo valores de entrada e saída a serem aprendidos. A cada programa são fornecidos os valores de entrada e confronta-se a sua resposta ao valor esperado de saída. A *aptidão* será proporcional à proximidade da resposta do programa ao valor de saída esperado. O operador de reprodução apenas seleciona um programa e o copia para a próxima geração sem sofrer nenhuma mudança em sua estrutura. As Figuras 3.6 e 3.7 mostram a aplicação do operador de recombinação (*crossover*) em duas funções selecionadas, que partilham informação genética e dão origem a duas novas funções diferentes.

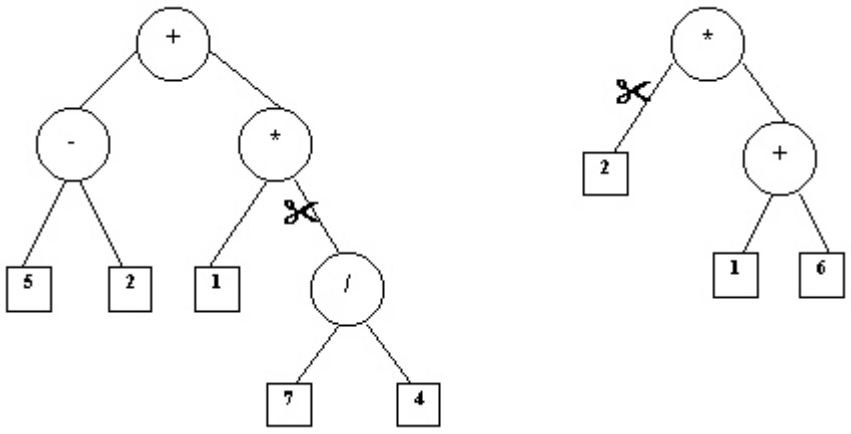

Figura 3.6 – *Crossover* na *PG* : seleção aleatória dos ramos que sofrerão o corte (SOUSA & ANDRADE, 1998).

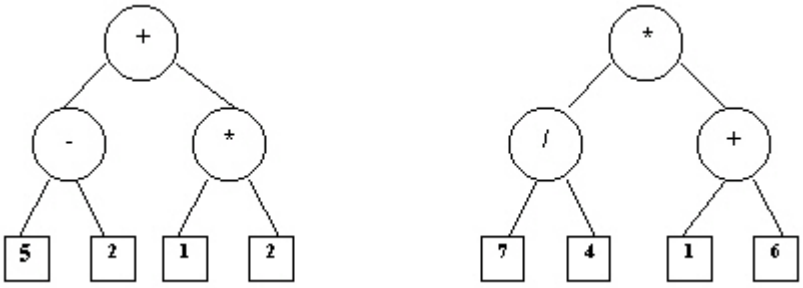

Figura 3.7 – *Crossover* na *PG* : funções resultantes (SOUSA & ANDRADE, 1998).

A mutação nem sempre é efetuada, pois depende de um valor que indica a probabilidade de existir mutação numa determinada geração. Quando é efetuada, uma função é escolhida aleatoriamente para sofrer mutação (Figura 3.8).

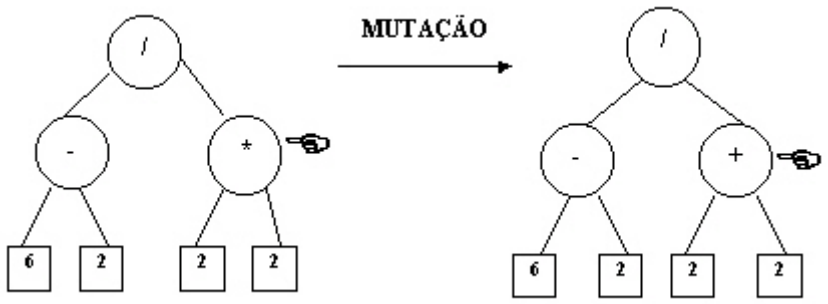

Figura 3.8 – Aplicação do operador de mutação na *PG* (SOUSA & ANDRADE, 1998).

3.4.1.6. Programação Evolutiva *(PE)*

De acordo com BÄCK et al. (1997) e MICHALEWICZ (1996), a PE surgiu originalmente como uma tentativa de criar inteligência artificial. O objetivo era desenvolver máquinas de estado finitas (MEF) para prever eventos com base em observações anteriores. Uma MEF é uma máquina abstrata que transforma uma seqüência de dados de entrada em uma seqüência de dados de saída. A transformação depende de certas regras de transição.

Os indivíduos são usualmente representados por vetores de números reais. Geralmente cada genitor gera um filho. A mutação ocorre tipicamente com probabilidade uniforme e é originalmente implementada como uma mudança

randômica (ou através de múltiplas mudanças) da descrição das MEF de acordo com cinco diferentes modificações:

- mudança de um dado de saída;
- mudança de uma regra de transição;
- inclusão de uma regra de transição;
- exclusão de uma regra de transição;
- mudança da regra de transição inicial.

Não é utilizada a recombinação.

O processo de seleção ocorre como uma série de torneios entre sub-grupos dentro da população. Cada indivíduo da população é avaliado contra t (obrigatoriamente $t > 1$ e usualmente $t \leq 10$) outros indivíduos escolhidos randomicamente da população. Para cada comparação é marcado um vencedor. Permanecem na população os μ indivíduos que tiveram o maior número de vitórias.

3.4.1.7. Estratégia Evolutiva *(EE)*

A primeira versão de $EE's$ foi $(1+1)-EE$, que empregava um esquema simples de seleção-mutação trabalhando em um único indivíduo que gera um único descendente através da mutação Gaussiana e ambos são submetidos ao processo de seleção, que elimina a solução mais pobre. Mais tarde, esta teoria evolui para $(\mu+1)-EE$, no qual uma população de μ indivíduos se recombina de maneira randômica para formar um descendente, que sofre mutação e em seguida, passa pelo processo de seleção.

Nas versões descritas acima, a convergência era lenta e a busca ponto a ponto era susceptível a estagnar em mínimos locais.

Mais tarde, visando sanar essas deficiências, desenvolveram-se outras versões, utilizando a estratégia denominada multi-membros, onde o tamanho da população é maior que um. Atualmente, os dois principais tipos são (COSTA & OLIVEIRA, 2002; BEYER et al., 2002; BÄCK et al., 1997):

- $(\mu+\lambda)-EE$

  Conhecida como estratégia soma, onde μ pais produzem λ filhos, sendo $\lambda > \mu$, gerando uma população de $\mu+\lambda$ indivíduos. Nesta estratégia, os

$\mu + \lambda$ indivíduos participam do processo de seleção, que determina os μ indivíduos que serão os pais da próxima geração.

- $(\mu, \lambda) - EE$

  Conhecida como estratégia vírgula, se difere da estratégia soma porque apenas os λ filhos participam do processo de seleção. Assim, o período de vida de cada indivíduo é limitado a apenas uma geração. Segundo CORTES & SAAVEDRA (2000), este tipo de estratégia tem bom desempenho em problemas onde o ponto ótimo é em função do tempo, ou onde a função é afetada por ruído.

Note também que ambas estratégias apresentadas são extremos da estratégia mais geral $(\mu, k, \lambda) - EE$, onde $1 \leq k \leq \infty$ representa o número máximo de gerações que um indivíduo pode permanecer na população.

Nas versões atuais, a descendência é obtida submetendo-se os indivíduos da geração a dois operadores: cruzamento e mutação. O cruzamento é feito de forma aleatória e a mutação é feita tipicamente através de uma perturbação Gaussiana de média nula e desvio padrão unitário, porém outros tipos de mutação são possíveis. Aplica-se também a idéia de auto-adaptação do parâmetro desvio padrão (σ) durante o processo evolutivo, o que é uma das características chaves do sucesso das estratégias evolutivas.

3.4.1.7.1. Distribuição Normal

Para utilizar um algoritmo de $EE's$ é necessário conhecer uma maneira de gerar variáveis aleatórias segundo uma distribuição normal ou gaussiana.

O modelo probabilístico citado acima é chamado Modelo Normal e suas origens remontam a Gauss em seus trabalhos sobre erros de observações astronômicas, por volta de 1810, daí o nome de distribuição Gaussiana para tal modelo (BUSSAD & MORETTIN, 2004).

De uma maneira geral, diz-se que uma variável aleatória (v.a.) β tem distribuição normal com média α e variância σ^2, onde $-\infty < \alpha < +\infty$ e $0 < \sigma^2 < \infty$, se sua função de densidade de probabilidade é dada por:

$$f(\beta, \alpha, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(\beta-\alpha)^2/2\sigma^2}, -\infty < \beta < \infty \qquad (3.3)$$

Podemos dizer que $\beta \sim N(\alpha, \sigma^2)$.

A Figura 3.9 ilustra uma curva normal, determinada por valores particulares de α e σ.

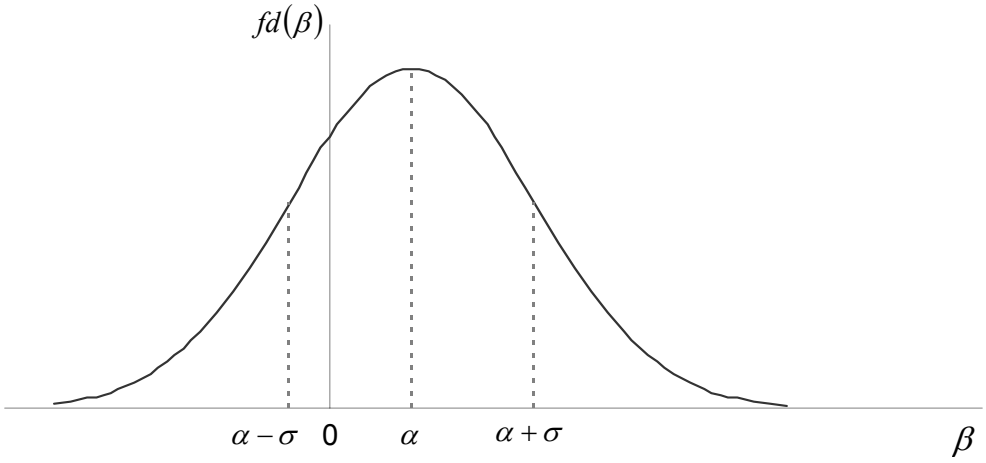

Figura 3.9 – Função de densidade de probabilidade de uma v.a. normal com média α e desvio padrão σ.

Quando $\alpha = 0$ e $\sigma = 1$, temos uma distribuição padrão ou reduzida.

Há vários métodos para gerar v.a. normais, mas uma observação importante é que basta gerar uma v.a. normal padrão, pois qualquer outra pode ser obtida desta. De fato, gerado um valor z_1 da v.a. $Z \sim N(0,1)$, para gerar um valor β_1 de uma v.a. $\beta \sim N(\alpha, \sigma^2)$ basta usar a transformação;

$$\beta_1 = \alpha + \sigma . z_1 \tag{3.4}$$

Um método eficiente para gerar v.a. com distribuição normal é o Método de Box-Müller (BUSSAD & MORETTIN, 2004). Nesse método são geradas duas v.a. normal padrão z_1 e z_2, independentes, e $N(0,1)$, a partir de duas v.a. com distribuição uniforme em [0,1], u_1 e u_2, como mostra as equações 3.5 e 3.6 (BUSSAD & MORETTIN, 2004) :

$$z_1 = \sqrt{-2 \log u_1} \cos(2\pi u_2) \tag{3.5}$$

$$z_2 = \sqrt{-2 \log u_1} sen(2\pi u_2) \tag{3.6}$$

A Figura 3.10 mostra o resultado da geração de números aleatórios usando a função *rand* da biblioteca padrão da linguagem C.

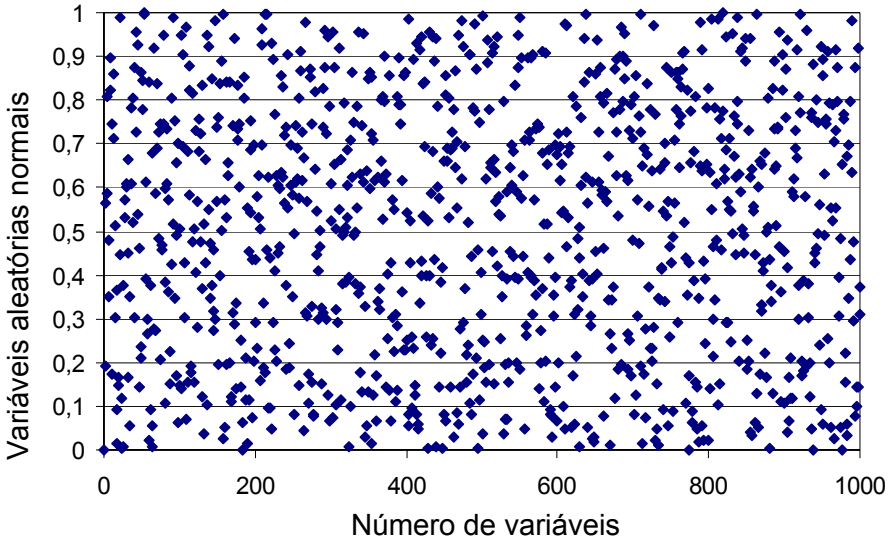

Figura 3.10 – Números gerados pela função *rand* da biblioteca da linguagem C.

Na Figura 3.11 é apresentado o resultado da geração de números aleatórios com distribuição normal a partir da variável aleatória uniforme gerada pela função *rand* da biblioteca da linguagem C.

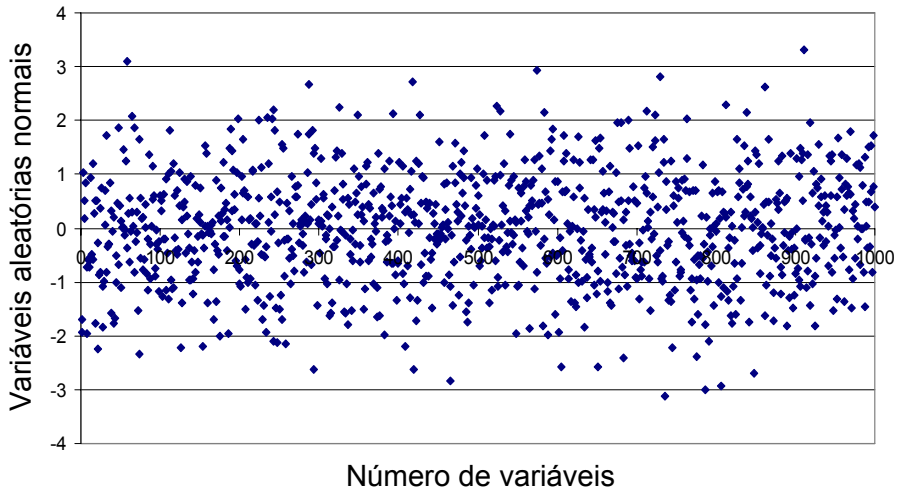

Figura 3.11– Números gerados pela transformação da v.a. uniforme em v.a. normal.

3.4.1.7.2. Algoritmo Padrão de *EE*

As componentes básicas de um Algoritmo Evolucionário quando aplicadas a um algoritmo de $EE's$ possuem características particulares, que estão detalhadas a seguir (CORTES & SAAVEDRA, 2000; EIBEN & SMITH,2003):

- Representação dos Indivíduos

  Nas $EE's$, cada indivíduo é representado por um par de vetores reais da forma $\boldsymbol{v} = (\boldsymbol{x}, \boldsymbol{\sigma})$, onde $\boldsymbol{x}$ representa um ponto de busca no espaço, ou seja, é o vetor das variáveis da função objetivo, e $\boldsymbol{\sigma}$ o vetor de desvio padrão associado.

- Inicialização da população

  A inicialização da população geralmente é feita de maneira muito simples, gerando aleatoriamente os indivíduos. Porém, pode-se utilizar alguma heurística para iniciar a população, tal como gerar indivíduos que sejam possíveis soluções do problema.

- Recombinação dos μ pais até gerar λ descendentes.

  Há inúmeras variações desse operador. Quanto ao número de genitores que participam da recombinação, ela pode ser chamada de recombinação de multi-pais, onde mais de dois indivíduos participam da geração de apenas um descendente, sendo $\rho (1 \leq \rho \leq \mu)$, onde ρ é o número de indivíduos que irão participar da recombinação para gerar um descendente. Normalmente, escolhe-se ρ=2 ou $\rho = \mu$ (recombinação global). Quanto as diferentes maneiras de recombinar os genitores, pode-se citar como exemplos típicos a recombinação discreta e a recombinação intermediária:

  - Recombinação discreta

    Um descendente é gerado a partir de dois ou mais genitores escolhidos randomicamente na população ancestral. As variáveis que irão formar o novo descendente são escolhidas randomicamente entre as variáveis dos genitores.

Para ilustrar esse processo é mostrado um exemplo onde dois indivíduos da população ancestral, $\boldsymbol{a} = (\boldsymbol{x}_a, \boldsymbol{\sigma}_a)$ e $\boldsymbol{b} = (\boldsymbol{x}_b, \boldsymbol{\sigma}_b)$, são escolhidos randomicamente e recombinados para formar um descendente, $\boldsymbol{v}' = (\boldsymbol{x}', \boldsymbol{\sigma}')$. A recombinação é feita gerando-se uma variável aleatória u com distribuição uniforme no intervalo de [0,1], amostrada individualmente para cada componente do vetor $\boldsymbol{v}'$.

$$u \leq 0.5 \quad \rightarrow \quad x_i' = x_{a,i}$$
$$u > 0.5 \quad \rightarrow \quad x_i' = x_{b,i}$$
$$u \leq 0.5 \quad \rightarrow \quad \sigma_i' = \sigma_{a,i}$$
$$u > 0.5 \quad \rightarrow \quad \sigma_i' = \sigma_{b,i}$$

Com i=1,...,n ; onde n é o número de variáveis da função objetivo.

- Recombinação Intermediária

A diferença da recombinação discreta é que as variáveis que irão formar o novo indivíduo são obtidas através da média aritmética das variáveis dos pais ao invés de realizar uma escolha randomica das variáveis. Sendo assim, usando o mesmo exemplo mostrado acimo, as variáveis do novo descendente poderiam ser obtidas da seguinte

$$x_i' = (x_{a,i} + x_{b,i})/2$$
$$\sigma_i' = (\sigma_{a,i} + \sigma_{b,i})/2$$

As vantagens e desvantagens da recombinação para uma função objetivo em particular devem ser notadas durante o desenvolvimento, pois não há uma recomendação generalizada para o uso deste operador.

- Mutação do desvio padrão e dos descendentes

Faz-se a mutação dos desvios padrões e, em seguida, a mutação dos descendentes seguindo as equações 3.7 e 3.8 (BÄCK & HAMMEL, 1994; BÄCK et al, 1997):

$$\sigma_i' = \sigma_i . exp(\tau' . N(0,1) + \tau . N_i(0,1)) \quad (3.7)$$

$$x_i' = x_i + N(0, \sigma_i') \quad (3.8)$$

onde:

i = 1,...,n; sendo n o número de variáveis da função objetivo;

$N(0,1)$ representa um número Gaussiano com média zero e desvio padrão unitário. Nota-se que esse número é o mesmo para todos os indivíduos quando multiplicado pelo fator τ' e, quando multiplicado por τ, deve ser obtido independentemente para cada valor de i. Os valores sugeridos para os parâmetros τ' e τ são mostrados nas equações 3.9 e 3.10, respectivamente.:

$$\tau' = (\sqrt{2n})^{-1} \tag{3.9}$$

$$\tau = \left(\sqrt{2\sqrt{n}}\right)^{-1} \tag{3.10}$$

Este esquema pode sofrer modificações. Uma opção é usar uma versão simplificada, onde é usado o mesmo desvio padrão para todas as variáveis da função objetivo.

É importante observar que os valores de τ e τ' dependem das características da função objetivo e os valores ótimos para estes parâmetros podem ser diferentes dos valores propostos.

Nota-se que na mutação dos descendentes o desvio padrão será diferente a cada geração, o que consiste no conceito de auto-adaptação.

- Seleção

  Avalia-se a *aptidão* dos genitores e descendentes, onde serão escolhidos os μ indivíduos com os melhores valores da *aptidão*, os quais serão os pais na próxima geração.

3.4.1.8. Comparação entre Estratégia Evolutiva e Algoritmo Genético

A partir do que foi visto nos itens anteriores, a Tabela 3.1 apresenta uma comparação entre Estratégia Evolutiva e Algoritmo Genético na sua forma padrão.

Tabela 3.1 – Comparação entre Estratégia Evolutiva e Algoritmo Genético

	EE	AG
Representação	Números reais	Binária
Seleção	Esquema de seleção determinístico: $(\mu + \lambda)$ ou (μ, λ).	Esquema de seleção probabilístico: A probabilidade de cada indivíduo permanecer na população é proporcional ao valor de sua *aptidão*.
Mutação	Principal operador	Operador de pequena importância e é aplicada através de inversão de bits.