Agoritmos Genéticos

Operadores e Populações

Separando os Operadores

- Até agora usamos somente um operador genético, o operador de crossover mais mutação de bit;
- A partir de agora dividiremo-lo em dois operadores separados (um para crossover e outro para mutação);
- Separamos com o intuito de obter maior controle sobre a operação de cada um deles;

Separando os Operadores

- Separando:
  - Podemos aumentar ou diminuir a incidência de cada um dos operadores sobre nossa população;
  - Cada operador receberá uma avaliação;
  - Para decidir qual será aplicado a cada instante rodaremos uma roleta viciada;
  - A seleção de indivíduos é feita depois da seleção do operador genético a ser aplicado, visto que o operador de mutação requer somente um indivíduo enquanto que o de crossover requer dois;
  - Normalmente o operador de crossover recebe uma probabilidade bem maior que o operador de mutação;
  - Normalmente a probabilidade associada ao operador de mutação é igual a 100%-pc, onde pc é a probabilidade do operador de crossover

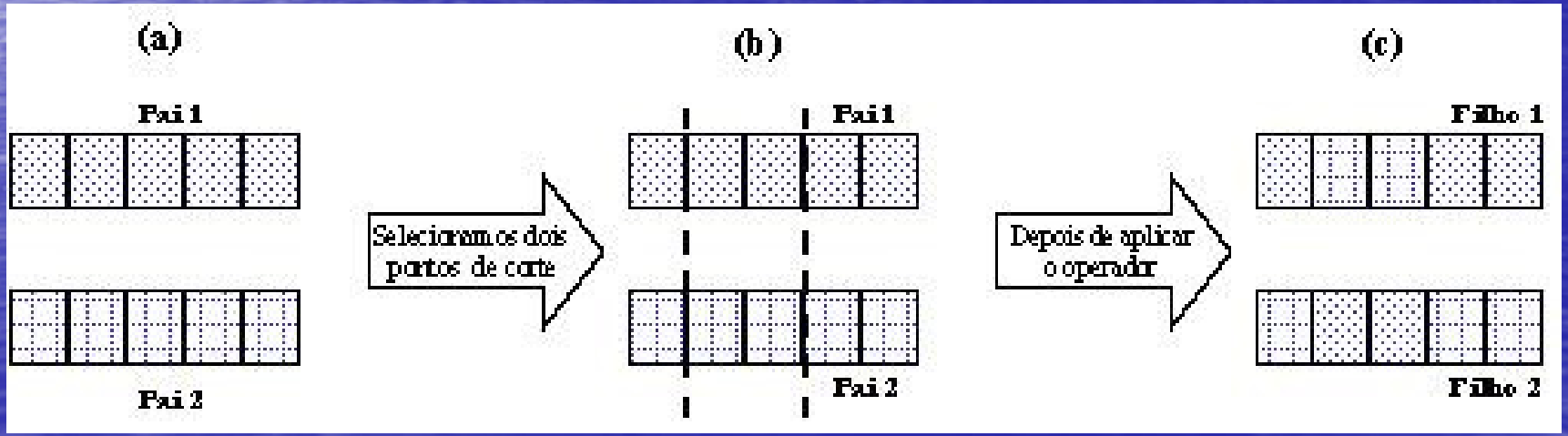
Outros Operadores

- Existem vários outros tipos de operadores diferentes;
- Entre os de crossover, podemos incluir:
  - Crossover de 2 pontos;
  - Crossover Uniforme;
  - Crossover de Maioria.
- Vamos discutir cada um deles.
- Eles são necessários, pois:
  - Existem dezenas de esquemas que o crossover de 1 só ponto não consegue preservar, como por exemplo 1******1.
  - Se não mudarmos o nosso operador de crossover, nosso GA ficará limitado na sua capacidade de processar esquemas

Crossover de Dois Pontos

- Funcionamento:
  - Sortearmos dois pontos de corte;
  - O primeiro filho será então formado pela parte do primeiro pai fora dos pontos de corte e pela parte do segundo pai entre os pontos de corte;
  - O segundo filho será formado pelas partes restantes.

- Graficamente:



Crossover de Dois Pontos

- Exemplo:
  - Suponha que temos dois pais de tamanho 10, dados respectivamente pelas strings *0101010101* e *1111000011*;
  - Sorteamos os pontos de corte 4 e 8;
  - Primeiro filho será dado, então, por:
    - a parte do primeiro pai até o ponto de corte 4 (*0101*);
    - a parte do segundo pai entre o ponto de corte 4 e o ponto de corte 8 (*0000*);
    - a parte do primeiro pai localizada após o ponto de corte 8 (*01*);
    - No final, o valor deste filho será *0101000001*.

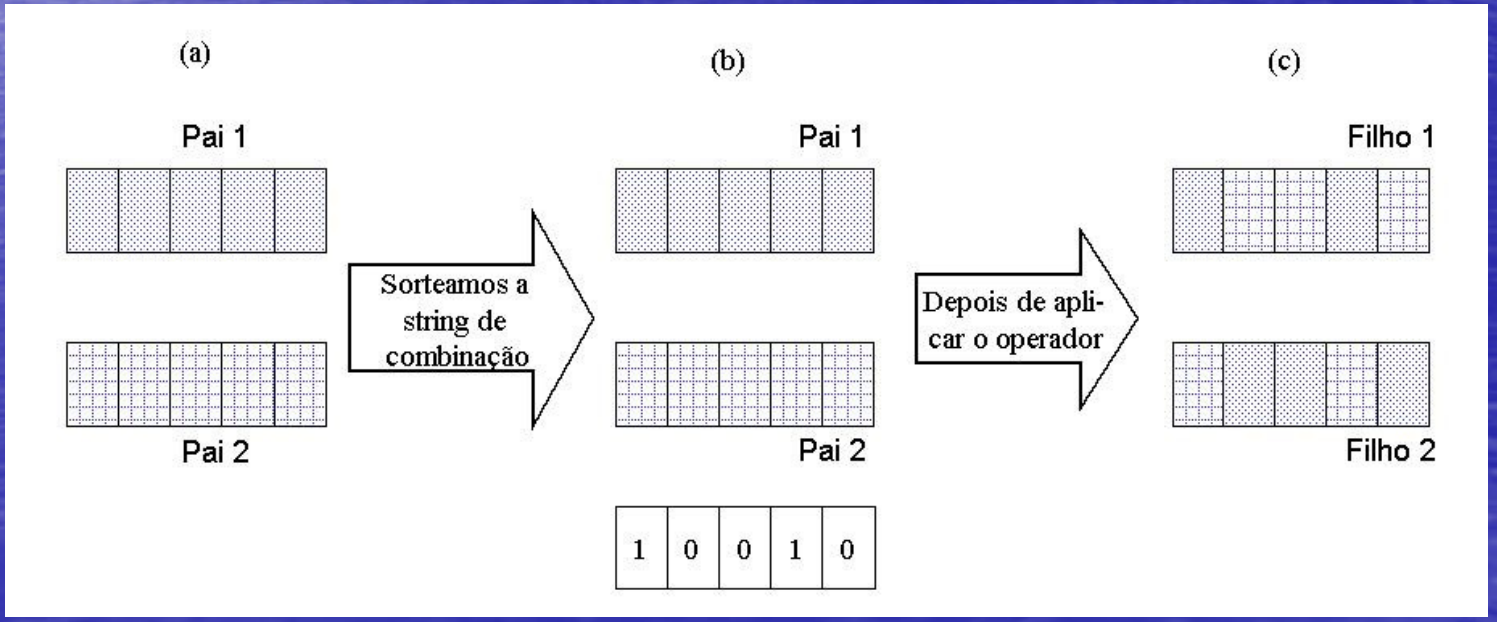
Crossover Uniforme

- O número de esquemas que podem ser efetivamente transferidos aos descendentes usando-se o crossover de dois pontos aumenta de forma considerável;

- Ele é ainda maior se usarmos o operador de crossover uniforme.

Crossover Uniforme

- Para cada gene é sorteado um número zero ou um.
- Se sorteamos um, primeiro filho recebe gene da posição corrente do primeiro pai e o outro, o do segundo pai.
- Se o valor sorteado for zero, as atribuições serão invertidas.
- Graficamente:



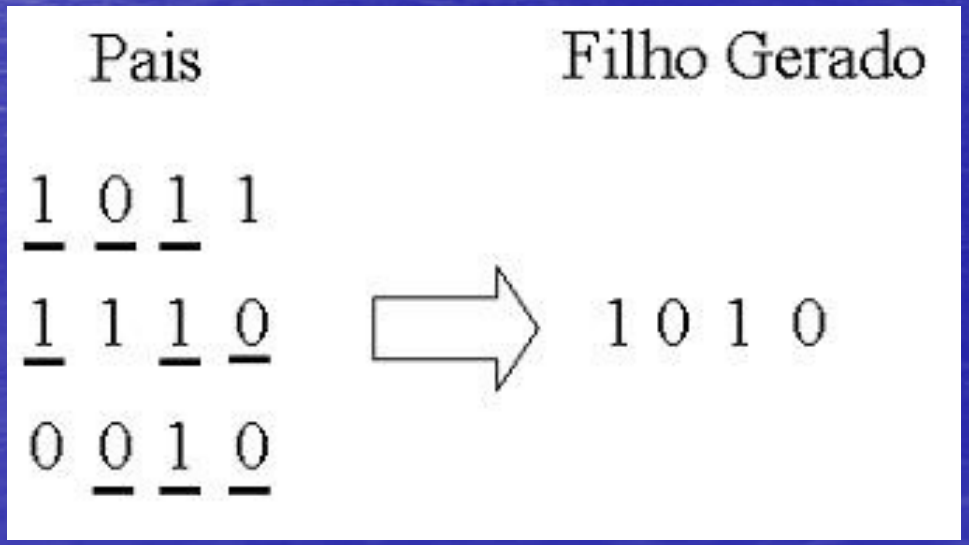
Crossover Uniforme

- Este operador tende a conservar esquemas longos com a mesma probabilidade que preserva esquemas de menor comprimento, desde que ambos tenham a mesma ordem.

- Devido ao fato de fazer um sorteio para cada posição, este crossover tem uma grande possibilidade de estragar todo e qualquer esquema, mas em média o seu desempenho é superior à dos seus antecessores.

Crossover Baseado em Maioria

- Não muito usado pois tende a fazer com que a convergência genética ocorra rapidamente.
- Operação básica:
  - sortear *n* pais
  - cada bit do filho seja igual ao valor da maioria dos pais selecionados.
- Graficamente:

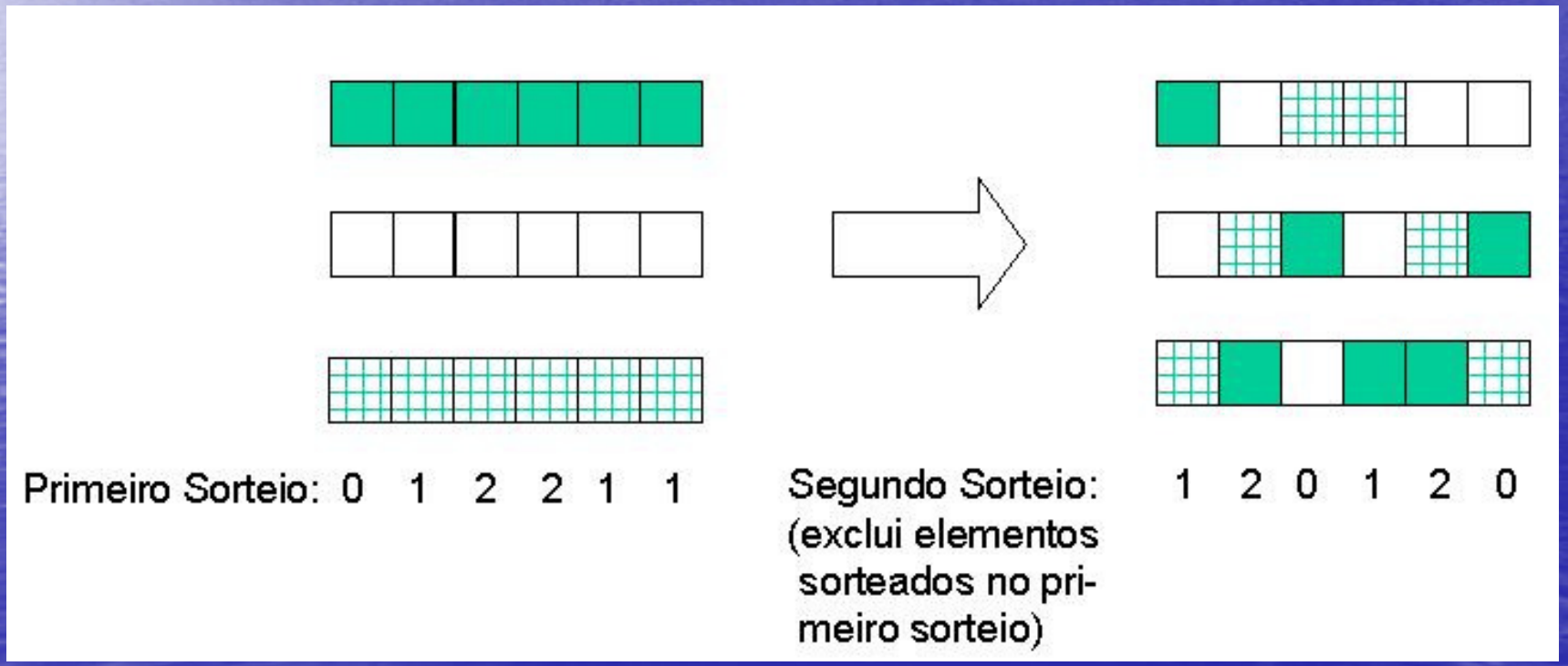


Crossover com mais de dois pais

- Não existe nenhum tipo de restrição formal que obrigue que o número de pais participando de uma operação de crossover seja igual a dois;
- Podemos usar quantos pais quisermos, bastando adaptar o operador utilizado;
- Por exemplo, se quisermos usar o crossover uniforme, selecionar 0,1 ou 2, para fazer a seleção do pai;
- Precisamos também de um segundo sorteio para determinar o pai que mandará o indivíduo para o segundo filho.
- Pode-se extrapolar para o caso em que temos n pais, quando teremos exatamente n-1 sorteios (o último filho é gerado com os "restos" não utilizados na composição de todos os seus irmãos)

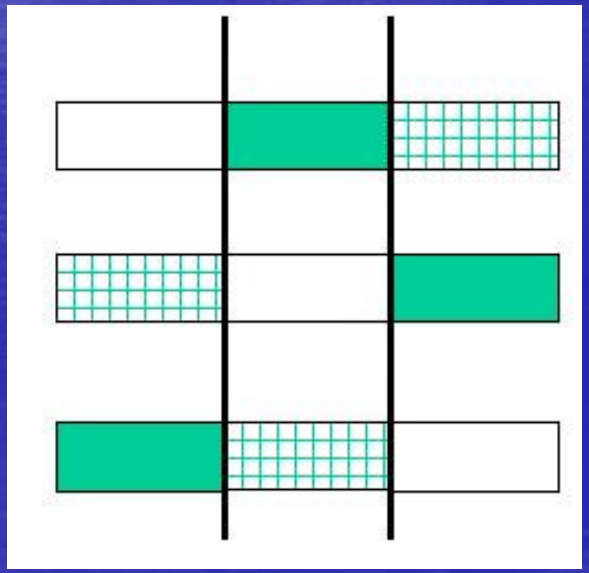
Crossover com mais de dois pais

- Graficamente:



Crossover com mais de dois pais

- Com *n* pais, vetamos crossovers de *k* pontos, onde $k<n-1$.
- Crossover de *k* pontos possui *k+1* blocos para combinar;
- Se temos mais pais do que *k+1*, alguns não participarão da composição dos filhos;
- Exemplo de uma operação possível para o crossover de 2 pontos usando três pais

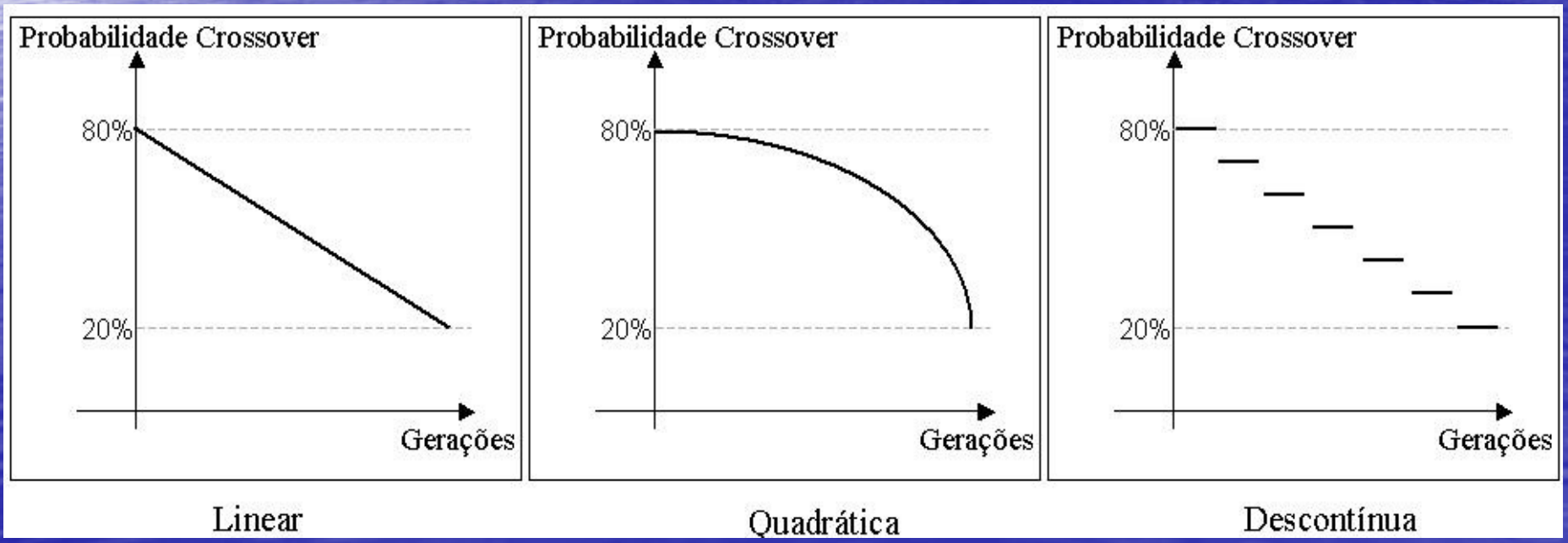


Operadores com Percentagens Variáveis

- Quando falávamos em dois operadores selecionados de forma separada, atribuíamos a cada um deles uma percentagem fixa e rodávamos uma roleta viciada de forma a escolher qual dos operadores seria aplicado sobre o indivíduo selecionado;
- Não há uma probabilidade que seja adequada para os dois operadores durante toda a execução do algoritmo;
- No início do GA, nós queremos executar muita reprodução e pouca mutação;
- Depois de um grande número de gerações, ocorre a convergência genética, tornando extremamente interessante que o operador de mutação seja escolhido mais freqüentemente.

Operadores com Percentagens Variáveis

- Precisaríamos que a probabilidade do operador de crossover fosse caindo com o decorrer do algoritmo e que a probabilidade do operador aumentasse;
- Podemos interpolar as probabilidades dos dois operadores;
- Há várias técnicas de interpolação candidatas:



Operador de Mutação Dirigida

- Pesquisadores resolvem o problema da convergência genética tornando a mutação mais provável que o crossover;
- Assim, há a tendência de que novamente venha a surgir uma variedade genética dentro da população de soluções.
- Todas as partes de cada uma das soluções têm igual probabilidade de serem modificadas, sem distinção.
- O problema pode estar concentrado no esquema dominante entre as melhores soluções;
- Se este esquema não for modificado de forma agressiva, pode ser que não cheguemos a lugar algum.

Operador de Mutação Dirigida

- Solução possível:
  - Criar um novo operador de mutação, que procurasse se concentrar neste esquema dominante;
  - Objetivo: criar variedade genética nos genes que nos interessam.
  - O operador só começa a agir depois de um grande número de gerações.
  - Quando ativado:
    - ele busca as *n* melhores soluções dentro da população padrão;
    - verifica qual é a bagagem cromossomial que elas têm em comum (usando operador XNOR);
    - pode-se considerar como comum cromossomos que estejam presentes apenas em uma maioria dos cromossomos.

Comentários sobre Operador de Mutação

- Operador de mutação é fundamental para um GA, pois garante a continuidade da existência de diversidade genética na população;
  - É uma heurística exploratória;
  - Se for sua probabilidade for baixa demais, população perderá diversidade rapidamente;
  - Se for alta demais, GA se comportará como uma *random walk.*
  - Solução razoável: utilizar uma taxa de mutação que varie com o desenrolar da evolução do algoritmo;
  - Com o steady state, pode-se aumentar taxa de mutação.

Comentários sobre Operador de Crossover

- Operador de crossover é o mais importante dos GAs;
- Somado ao módulo de seleção proporcional à avaliação de um indivíduo, é o responsável pelo fato de um algoritmo genético não poder ser comparado a uma busca aleatória;
- Historicamente, o operador de crossover tem recebido uma percentagem de escolha muito alta;
- Escolha do tipo de operador utilizado (representação binária):
  - Crossover uniforme é o mais poderoso
  - Pode criar o mesmo tipo de soluções que os crossovers de um e de vários pontos, além de novas combinações que estes tipos não são capazes de criar.
  - Ele também é o mais destrutivo, sendo o crossover que mais separa elementos de um esquema interessante, especialmente se estes elementos fundamentais são adjacentes.

Comentários sobre Operador de Crossover

- Idéia interessante para minimizar poder destrutivo:
  - modificar a probabilidade de selecionar um gene de cada um dos pais
  - aumentando a probabilidade de selecionar o gene *i* do pai *k* se o gene *i-1* foi selecionado deste pai.
  - Esta solução diminui a probabilidade de romper esquemas que contenham elementos adjacentes, sem evitar que o crossover uniforme gere soluções livremente.
- Crossover de um ponto, possui uma característica denominada de preconceito positional (*positional bias*):
  - assume que esquemas curtos e de baixa ordem são os blocos funcionais básicos dos cromossomos;

Comentários sobre Operador de Crossover

- Existem vários trabalhos que evitam ter que escolher entre os operadores de crossover;
- Acrescenta-se uma roleta adicional ao processo de reprodução;
- Roleta serve para determinar qual operador de crossover será usado para aquele par de pais.
- Esta estratégia tenta combinar as características exploratórias mais agressivas do operador de crossover uniforme com o conservadorismo do crossover de um ponto.
- Introduz-se mais um parâmetro que deve ser controlado pelo desenvolvedor de GA, que é o conjunto de probabilidades de cada um dos tipos de crossover.

Outros Módulos de População

- Até agora, nosso módulo de população teve um comportamento extremamente simples
  - Todos os indivíduos da atual geração eram eliminados;
  - Depois, substituíamos um igual número de filhos.
- Entretanto, este não é o único comportamento possível:
  - Existem várias alternativas;
  - Estas permitem que exploremos melhor as qualidades da geração atual
  - Assim, podemos aproveitá-las para a melhoria da próxima geração.

Tamanho da População

- O desempenho do algoritmo genético é extremamente sensível ao tamanho da população;
- Caso este número seja pequeno demais, não haverá espaço para termos variedade genética:
  - Pode ser que nosso algoritmo seja incapaz de achar boas soluções;
- Caso este número seja grande demais, o algoritmo demorará demais:
  - Poderemos estar nos aproximando de uma busca exaustiva.

Tamanho da População

- A maioria dos trabalhos publicados tem uma fixação quase fetichista no número 100;
- Não existe número mágico!
- Aplique um pouco de raciocínio sobre o problema que se está tentando resolver...

Tamanho da População

- A capacidade de armazenamento de estados de um GA é exponencialmente proporcional ao número de indivíduos presentes;
- Há um limite superior para o tamanho da população onde ainda verifica-se melhora na performance conforme aumenta-se o tamanho da população;
- Quanto maior a população, mais tempo o GA demorará para processar cada geração e mais demoraremos para conseguir uma resposta

Não devemos aumentar o tamanho da população indiscriminadamente

Tamanho da População

- O número de indivíduos avaliados em uma execução do GA é igual ao número de indivíduos na população vezes o número de rodadas que o algoritmo irá executar;
- Veja se este número total é um percentual alto do espaço de busca:
  - Se a percentagem de soluções avaliadas for muito alta, pode-se considerar alguma heurística informada como técnica alternativa de resolução do problema.
- A dificuldade da função de avaliação deve ser um fator que afete a escolha destes dois parâmetros:
  - Se a função de avaliação for multi-modal e enganadora, o número de avaliações deve crescer;
  - Podemos aumentar o tamanho da população ou o número de gerações – ou até mesmo os dois!!!

Tamanho da População

- Tentativa inicial razoável para o número de indivíduos dentro da sua população:
  - 40*número de características em seu cromossomo
- É só uma idéia inicial, extremamente imprecisa...
- Talvez seja melhor usar uma estratégia adaptativa para o tamanho da população!
- Vamos discutir populações de tamanho variável...

Populações de Tamanho Variável

- Estratégia 1: definição de uma expectativa de vida para cada indivíduo:
  - Esta expectativa é proporcional à qualidade do indivíduo;
  - O tamanho da população possa crescer caso a avaliação de todos os indivíduos for muito boa;
  - Neste caso, estes sobreviverão por muitas gerações, além de gerar filhos que também irão compor a população.

Populações de Tamanho Variável

- O número de filhos gerados a cada geração é dado por:

$$\rho * P(t)$$

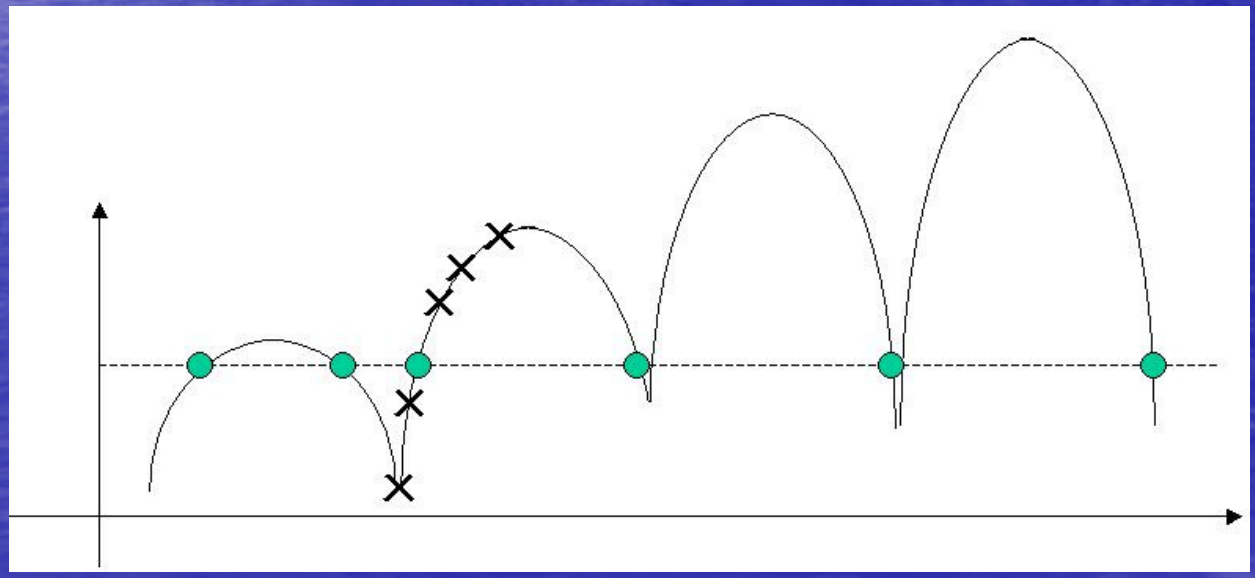
- Como a cada instante podemos gerar mais filhos do que o número de "mortos" da geração anterior, a população pode aumentar
- Ela pode diminuir se o oposto ocorrer!
- Não tem uma pressão seletiva forte sobre os indivídos:
  - Eles "morrem" quando atingem a "velhice"

População de Tamanho Variável

- Estratéga 2: aumentar o tamanho da população se:
  - está havendo convergência genética
  - ainda não chegamos perto da performance desejada.
- Problema: determinar quando a convergência genética aconteceu.
  - Não é uma tarefa simples!

População de Tamanho Variável

- O cálculo da variabilidade deve ser feito com base no genótipo dos indivíduos, e não com base na função de avaliação
- Indivíduos muito diferentes podem ter funções de avaliação muito parecidas
- Exemplo:



População de Tamanho Variável

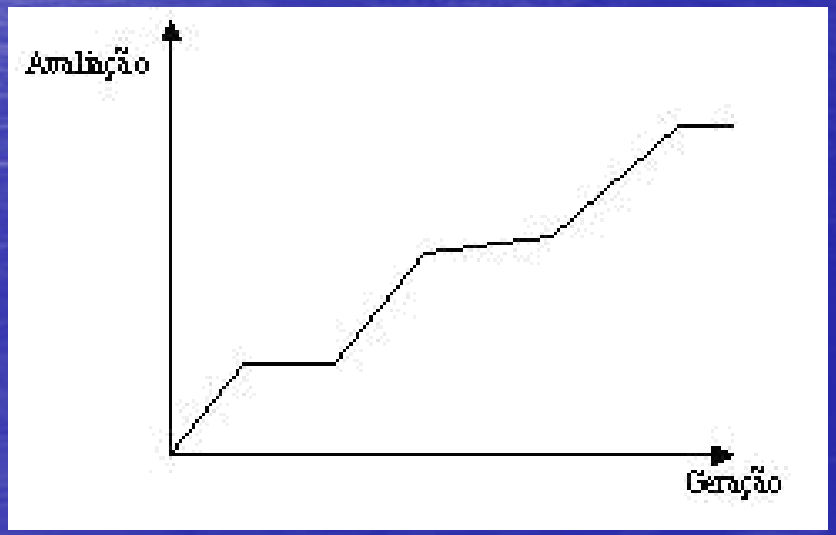
- Estratégia 3: PRoFIGA
- *Population Resizing on Fitness Improvement GA*
- Idéia:
  - aumentar o tamanho da população por um fator $\rho+$ caso a melhor avaliação tenha melhorado na última geração ou caso esta não tenha melhorado nas últimas k gerações.
  - Caso nenhuma destas duas condições seja satisfeita, então a população é diminuída por um fator $\rho-$

Elitismo

- Pequena alteração no módulo de população que quase não altera o tempo de processamento;
- Garante que o desempenho do GA sempre cresce com o decorrer das gerações;
- Idéia básica:
  - Os *n* melhores indivíduos de cada geração não devem "morrer";
  - Estes devem passar para a próxima geração para garantir que seus genomas sejam preservados.

Elitismo

- Manutenção do melhor indivíduo da geração *t* na população da geração *t+1* garante que o melhor indivíduo da geração *t+1* é pelo menos igual que o melhor indivíduo da geração k;
- Curva típica de desempenho:



Elitismo

- Este pequeno ato, apesar de sua simplicidade, normalmente colabora de forma dramática para a melhoria do desempenho de uma execução de um GA;

- Motivo:
  - mantemos dentro da população os esquemas responsáveis pelas boas avaliações das melhores soluções;
  - aumenta o componente de memória de nosso algoritmo genético.

Steady State

- Até agora toda uma geração nasce ao mesmo tempo enquanto que a população anterior morre também toda de uma vez e é substituída por esta novos indivíduos;
- No "mundo real", os indivíduos nascem aos poucos, os mais velhos morrem de forma lenta e há interação entre as gerações;
- Indivíduos da geração $t+1$ podem procriar com indivíduos da geração t (sexismo e preconceitos de idade à parte)

Steady State

- Steady state busca reproduzir este tipo de característica natural das populações biológicas;
- Ao invés de criarmos uma população completa de uma só vez, vamos criando os "filhos" um a um (ou dois a dois, por ser mais conveniente para o operador de crossover);
- Substituimos os piores "pais" por estes novos indivíduos.

Steady State

- Algumas implementações de GA substituem os pais aleatoriamente em vez de substituir necessariamente os piores pais;
  - Aumenta as chances de preservar piores;
  - Manter somente os melhores faz com que tenhamos uma tendência maior à convergência genética;
  - Alternativa: usar uma roleta viciada invertida para selecionar os pais moribundos.

Steady State

- Conceito de geração dentro do nosso GA fica muito difuso, quase inexistente:
  - Pode haver reprodução entre indivíduos recém criados e outros da geração anterior;
  - Pode haver até mesmo "incesto", se não mantivermos o registro de quem são os pais de um indivíduo.
- Há maior dominação dos melhores esquemas:
  - normalmente faz com que a população convirja mais rapidamente
  - especialmente se eliminarmos sempre os piores elementos a cada operação realizada

Steady State

- Questões interessantes:
  - Devemos garantir uma sobrevida mínima para cada indivíduo recém-criado?
  - Devemos evitar "incesto"?
    - Alternativa: verificar semelhança entre pais selecionados.

Steady State sem Duplicatas

- Usada para evitar que a convergência seja rápida demais;
- Diferença: se o indivíduo gerado for idêntico a algum já presente na população ele é descartado;
- Para cada operação de crossover ou de mutação realizada precisamos verificar se os filhos resultantes já estão presentes na população;
- Tende a conseguir melhores resultados, às custas de um *overhead* grande .

Estratégia ($\mu+\lambda$)

- Idéia: população da próxima geração é selecionada a partir dos melhores indivíduos existentes, tanto na população corrente quanto naqueles filhos gerados pela aplicação dos operadores genéticos.

- Conhecido como sendo do tipo ($\mu+\lambda$):
  - existem μ membros na população original;
  - existem λ filhos;
  - geralmente, $\mu<\lambda$.

- Todos competem entre si!

Estratégia ($\mu+\lambda$)

- Competição usualmente é feita através da escolha dos indivíduos melhores avaliados, mas isto não é obrigatório;
- Pode-se tentar usar alguma métrica adicional em que a diversidade da população seja mantida;
- Assim, procuramos manter na população indivíduos que podem ajudar a explorar o espaço de busca de forma mais eficiente;
- É uma versão "avançada" do elitismo.
- Corre-se o risco de haver uma convergência genética mais rápida.