UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ – UTFPR

Programa de Pós-Graduação em Engenharia e Informática – CPGEI
Laboratório de Bioinformática e Inteligência Computacional-LABIC, Curitiba

Computação Evolucionária

Prof. Heitor Silvério Lopes
hslopes@utfpr.edu.br

Class2 (2020)

Fundamentos teóricos de Algoritmos Genéticos

- Terminologia, Biologia
- Definição formal e operação
- Critérios de parada
- Codificação e mapeamento genótipo/fenótipo
- Função de objetivo e função *fitness*
- Restrições e penalidades
- AG canônico
- Métodos de seleção e elistismo
- Operadores genéticos para representação binária, inteira e real
- Convergência, exploração e diversidade genética
- Controle da pressão seletiva: escalonamento de *fitness*, truncagem, *generation gap*
- Epistasia e decepção
- Nichos e espécies, fator de *crowding*

Terminologia

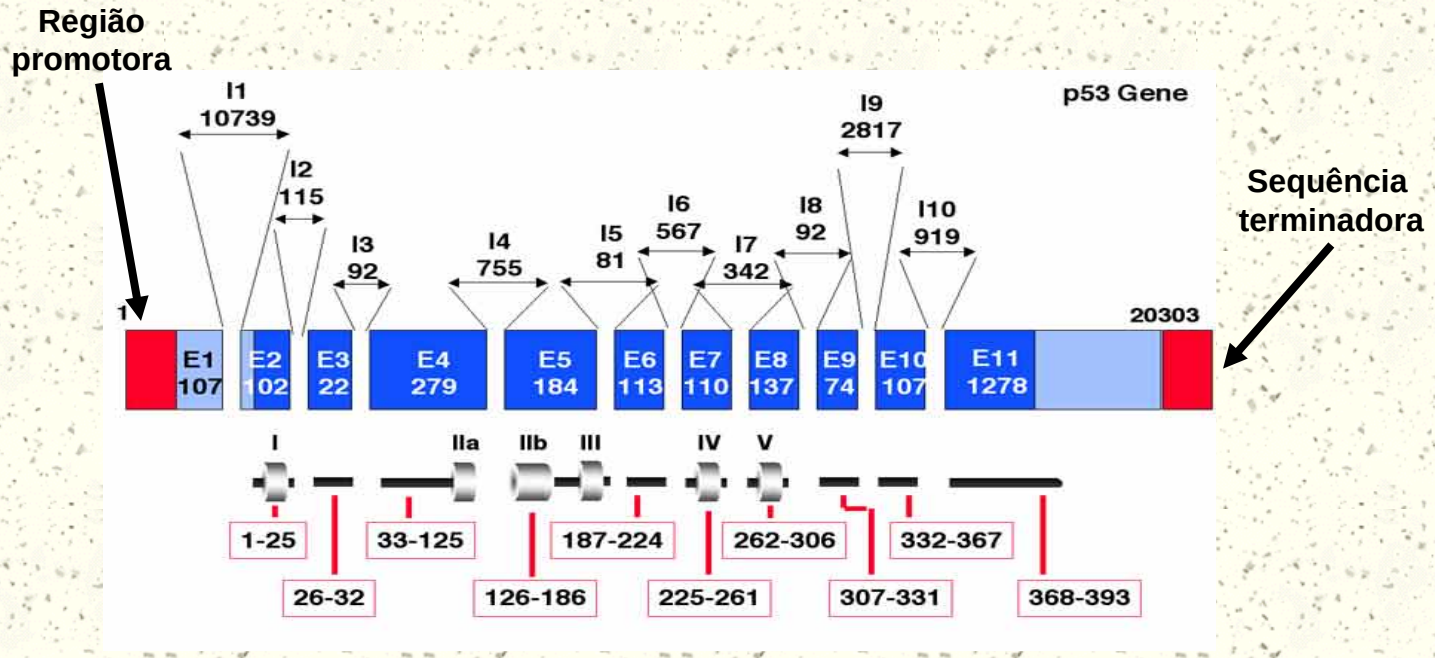
- Influência da Biologia (Genética e Ecologia)
- Relembrando...
  - Um organismo é definido pelos seus cromossomos (*Homo sapiens*: 22 pares, mais XX/XY)
  - Cada cromossomo tem inúmeros genes (*H.sapiens* > 50.000 genes no total) e sequências inter-gênicas
  - Cada gene contém exons e outras sequências não-codificantes (introns, região promotora, sequência terminadora)
  - Parte dos exons quando ligados sequencialmente e tomados em triplas serão os códons que codificam uma proteína com função biológica.

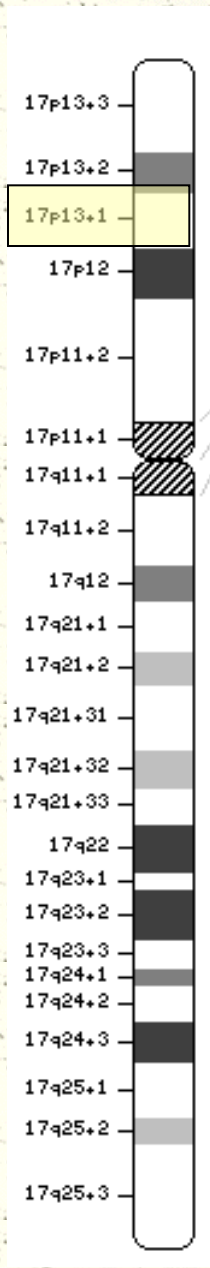
Terminologia: Biologia ←→ AG

Biologia	Algoritmos Genéticos
Cromossomo	*String*
Locus	Posição no *string*
Gene	Caracter, número ou bit
Genótipo	Estrutura, parâmetros (geralmente = *string* - monoploidia)
Fenótipo	Solução, ponto ou indivíduo
Epistasia	Interferência entre genes
Ploidia	Relativo ao número de pares de genes no genótipo de um organismo (monoploidia,diploidia...)

"Anatomia" de um gene

- Um cromossomo é composto de genes e de sequências intergênicas (sem função conhecida).
- Em AGs, todo o conteúdo do cromossomo é "codificante".
- Existem parcos estudos a respeito do uso de introns em AGs e outras técnicas de CE.





Definição formal de parâmetros

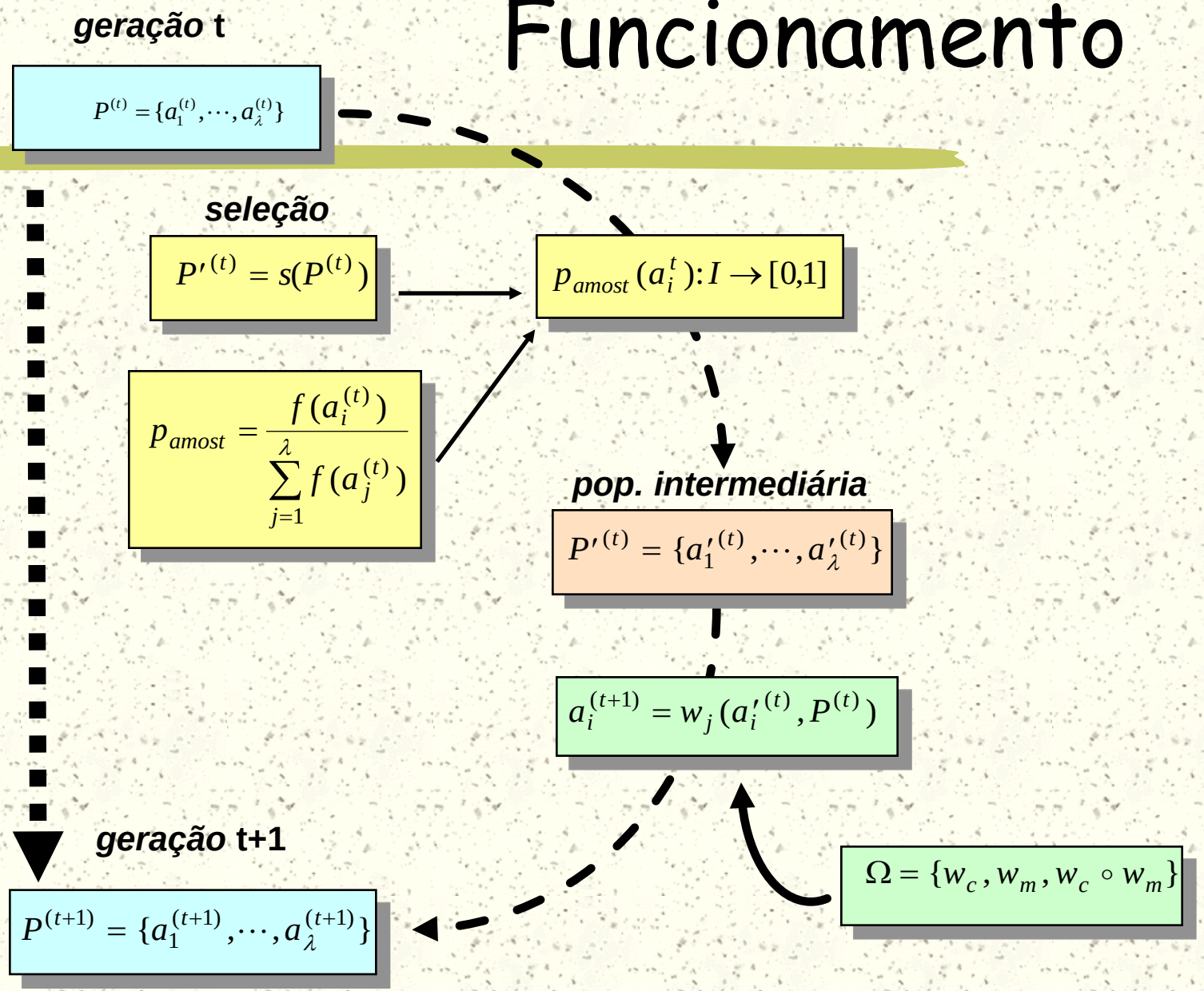
- Baseado em Hoffmeister e Bäck (1990)
- Definição de um AG simples como uma 8-tupla:

$$GA = \{P^0, \lambda, L, s, \rho, \Omega, f, t\}$$

Definição formal de parâmetros

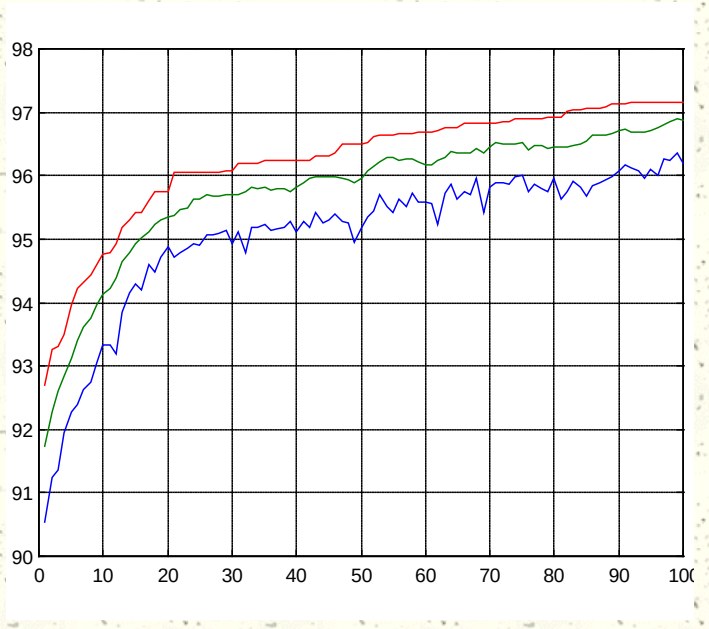
Definição do parâmetro	significado
$P^0=(a^0_1,\ldots,a^0_\lambda) \in I^\lambda,\ I=\{0,1\}^L$	população inicial
$\lambda \in \aleph$	tamanho da população
$L \in \aleph$	tamanho de cada *string*
$s: I^\lambda \rightarrow I^\lambda$	método de seleção
$\rho : I \rightarrow \Omega$	função que determina o operador
$\Omega \subseteq \{\omega: I \times I^\lambda \rightarrow \mathbf{Prob} \rightarrow I\}$	conjunto de operadores genéticos
$f : I \rightarrow \Re$	função de *fitness*
$t : I^\lambda \rightarrow \{0,1\}$	critério de término

Funcionamento

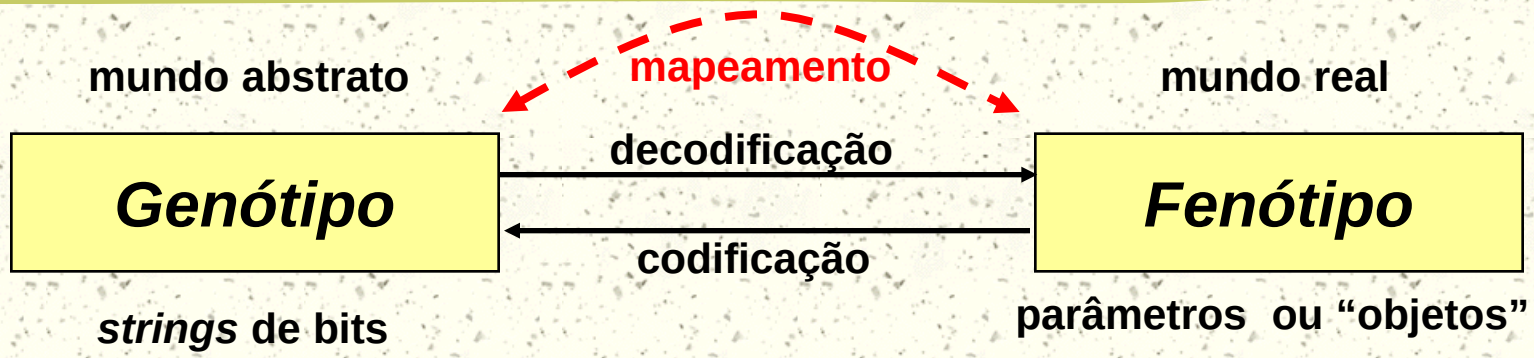


Critérios de parada

- Quando terminar a busca com AG, isto é, $t : I^{\lambda} \rightarrow 1$?
  - Parar após um período determinado de tempo ou número de gerações máximo
  - Parar quando não houver melhora significativa do máximo
  - Parar quando encontrar uma solução melhor do que outra existente (segundo algum critério)
  - Parar quando o ótimo for atingido
  - Parar quando a média de *fitness* da população estiver próxima ao mínimo ou máximo *****



Mapeamento genótipo X fenótipo



Princípios da Codificação Importante!

- A codificação é um elemento crítico na aplicação de AG's em problemas práticos.
- Um problema mal codificado pode impedir que a convergência do AG ou a obtenção de uma boa solução.
- Princípios da codificação:
  - Princípio dos blocos construtivos significativos
    - Como uma regra prática útil, sempre se deve colocar parâmetros importantes ou que tenham alguma inter-relação, juntos ou próximos no *string*, de modo a constituir blocos construtivos úteis.
  - Princípio dos alfabetos mínimos:
    - O usuário deverá selecionar o menor alfabeto que permita a expressão natural do problema.
    - Para a maioria dos casos a representação é simples e direta, através do mapeamento das variáveis do problema em um *string* binário, onde cada variável é digitalizada com o número de bits que for necessário para alcançar a precisão desejada do problema.

Codificações

- Binária
  - *Default*, para otimização discreta, não combinatorial ou otimização contínua discretizada
- Inteira
  - Para problemas combinatoriais
- Reais
  - Para otimização em espaços contínuos
- Importante: cada tipo codificação exige operadores específicos !!!

Codificação binária

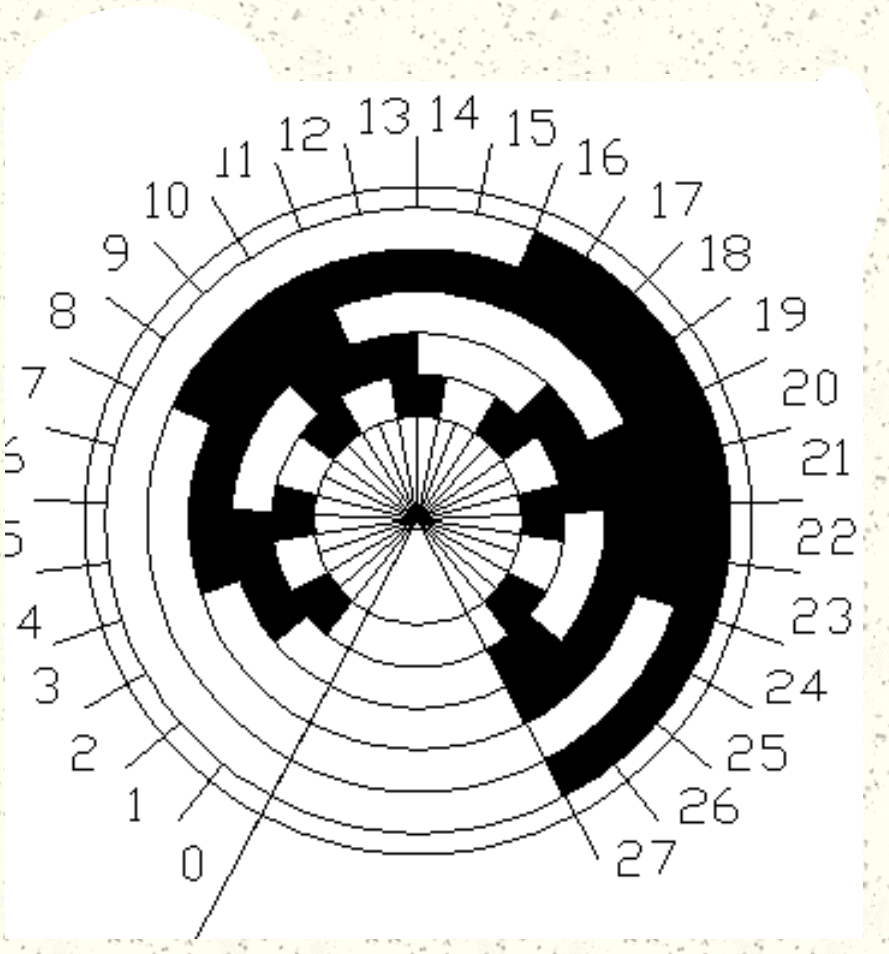
- Codificação em AGs:
  - binário natural ou código de Gray
- Codificação de múltiplos parâmetros:
  - cada variável = um gene
  - concatenação de vários parâmetros em um único *string*
- Codificação de números reais em binário:
  - Não é usual em AG
  - Quantização ou "discretização": mapeamento em um intervalo binário finito
  - número de bits proporcional à precisão (arbitrária)

$$precisão = \frac{X_{\max} - X_{\min}}{2^L}$$

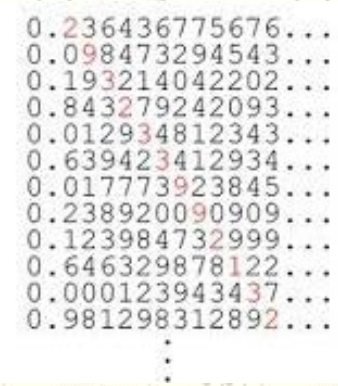
Alternativa ao binário natural: Código de Gray

- Muda um bit de um número para outro: transição "mais suave".

Decimal	Binário	Gray
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

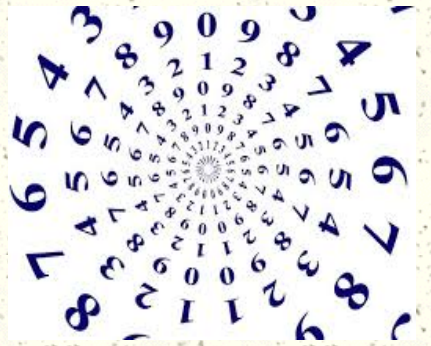


Codificação com números reais



- Quando se tem muitas variáveis e ao mesmo tempo é necessário alta precisão:
  - Cromossomos muito longos levam a espaços de busca intratáveis. Exemplo:
    - 100 variáveis, cada uma no intervalo [-500,500]
    - precisão 6 dígitos decimais
    - 30 bits por variável
    - espaço de busca: $2^{3000} \approx 10^{1000}$
- Codificação real: vantagens e desvantagens:
  - Processamento mais rápido (não tem decodificação)
  - Exige operadores específicos
    - Mutação: aleatória ou incremental
    - *Crossover* aritmético

Codificação com números inteiros

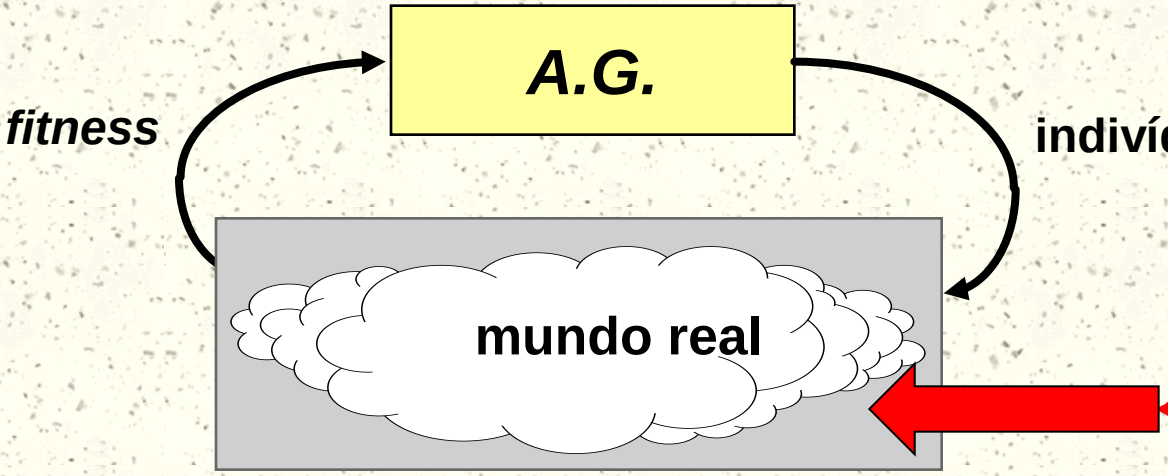


- Alternativas para permutações:
  - Utilizar números inteiros
  - Utilizar um alfabeto restrito
  - Utilizar binário (ou Gray) e decodificar adequadamente
- Implica em operadores genéticos especialmente adaptados para preservar a integridade do cromossomo

Função de *fitness*

Importante!

- É o ponto mais crítico de uma aplicação real.
- Também conhecida como função de "adequabilidade" ou "adaptabilidade"
- Abordagem "caixa-preta"



função de fitness:

- decodificação
- ajuste de escala
- manipulação de restrições
- aplicação de penalidades
- avaliação função objetivo

Função objetivo X Função de *fitness*

- Mapeamento $f_{objetivo} \longrightarrow f_{fitness}$
- Por definição, $f_{fitness}$ é não-negativa e desejavelmente deve ser normalizada num intervalo conhecido.
- A função objetivo pode envolver um custo ou um lucro
- Minimização de custo:

$$f(x) = \begin{cases} C_{max} - custo(x) & \text{quando } custo(x) < C_{max} \\ 0 & \text{qualquer outro caso} \end{cases}$$

onde C_{max} é o pior caso conhecido

- Maximização de lucro:

$$f(x) = \begin{cases} ganho(x) + C_{min} & \text{quando } ganho(x) + C_{min} > 0 \\ 0 & \text{qualquer outro caso} \end{cases}$$

onde C_{min} é o pior caso conhecido

Restrições (*Constraints*)

- Muitas aplicações reais exigem que a função objetivo seja submetida a certas restrições:

$$\begin{aligned} &\text{minimizar} \quad g(\mathbf{x}) \\ &\text{sujeito a} \quad h_i(\mathbf{x}) \geq 0 \qquad i = 1,2,\ldots,n \\ &\text{onde} \quad \mathbf{x} = x_1, x_2, \ldots, x_m \end{aligned}$$

- As restrições limitam os valores possíveis das variáveis do problema
- As restrições afetam diretamente a codificação do problema

Satisfação de restrições Importante!

- Durante a execução do algoritmo podem surgir soluções que não satisfazem as restrições
- É necessário alguma metodologia para satisfazer as restrições
- Opções:
  1. Desprezar soluções inválidas
  2. Fazer uma codificação especial, adaptada para cada variável de cada problema, que não permita soluções inválidas *****
  3. Permitir uma codificação mais flexível e aplicar penalidades no caso de violação de restrições. *****
  4. Descobrir, para cada variável, como cada restrição é violada e "consertar" a solução.

Aplicação de penalidades

- A aplicação de penalidades à função objetivo foi proposta por Goldberg (1989) e transforma o problema numa forma *unconstrained:*

$$\text{minimizar} \quad g(\mathbf{x}) + r\sum_{i=1}^{n}\Phi[h_i(\mathbf{x})]$$

onde: Φ = função de penalidade

r = coeficiente de penalidade

$\mathbf{x} = x_1, x_2, \ldots, x_m$

- O coeficiente de penalidade (*r*) pondera o quão importante é a violação de restrições para o problema.
- A função Φ é sugerida como o quadrado da violação da restrição.
- O conjunto das penalidades deve ser normalizado na mesma faixa de valores da normalização da função objetivo.

Métodos de Seleção

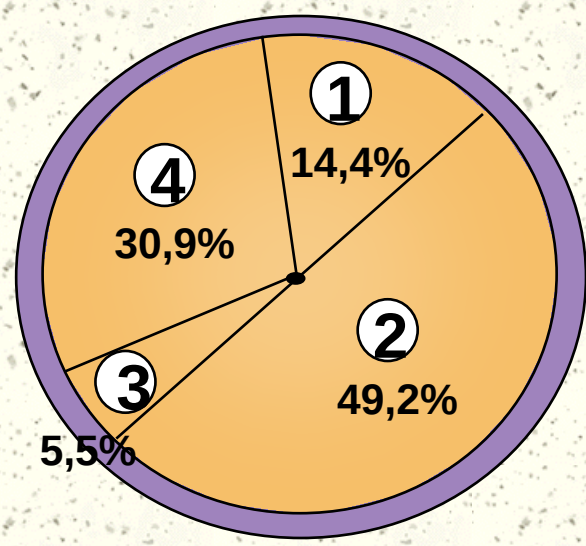
- Seleção – **NÃO** é operador
- A seleção é um processo executado a priori a fim de gerar indivíduos sobre os quais serão aplicados os operadores genéticos.
- É na seleção onde se evidencia o processo de seleção natural de Darwin.
- Os indivíduos mais bem adaptados ao ambiente têm maior probabilidade de se reproduzir e passar seu material genético para os descendentes

Seleção proporcional

- Método "clássico" e mais simples
- Também é conhecido como *Roulette wheel* ou roleta
- É um método muito ineficiente pois induz a convergência para máximos locais.

i	*string*	*fitness*	fit.rel.
1	01101	169	14,4%
2	11000	576	49,2%
3	01000	64	5,5%
4	10011	361	30,9%
		1170	100%

$$p_{seleção} = \frac{f(a_i^{(t)})}{\sum_{j=1}^{\lambda} f(a_j^{(t)})}$$

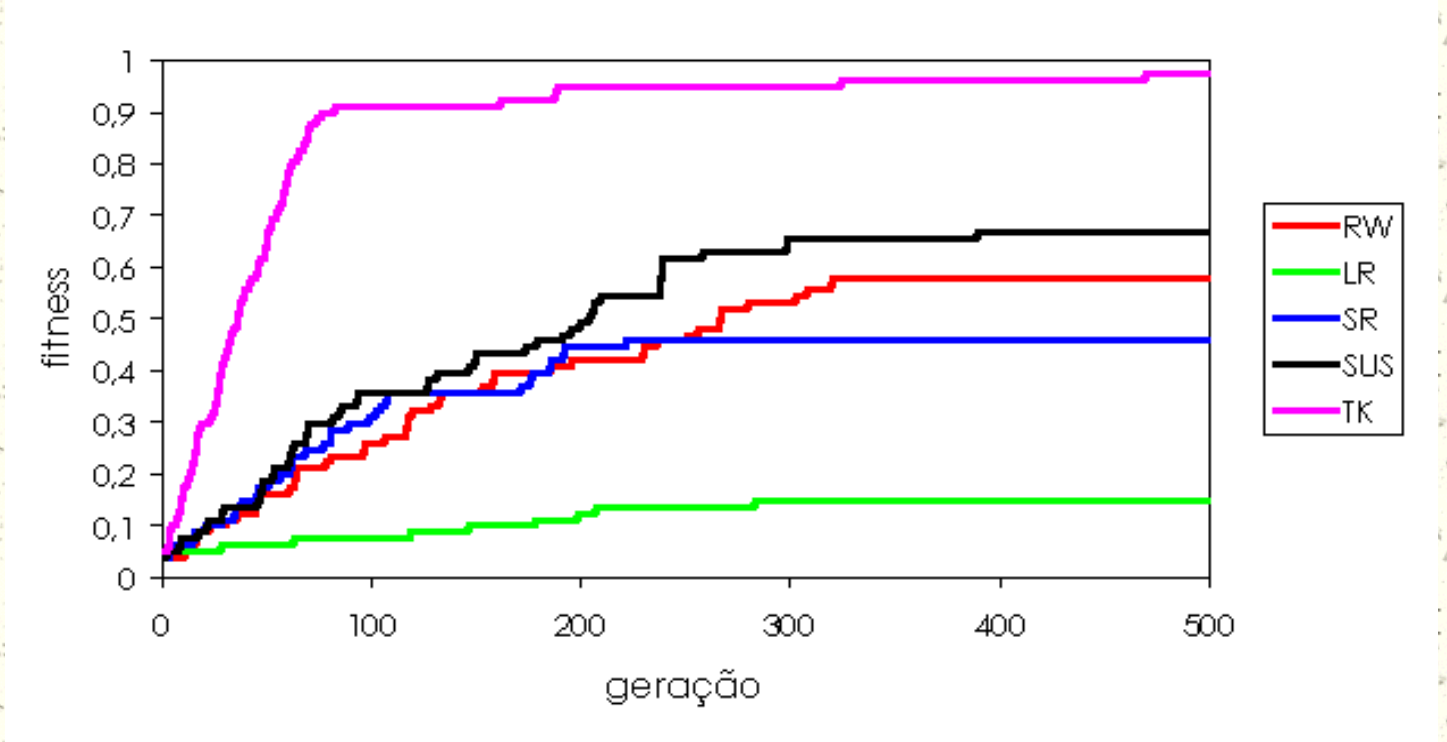


Outros métodos de seleção

- Ordenamento linear (*linear ranking*)
- Ordenamento uniforme (μ,λ) (*uniform ranking*)
- Amostragem estocástica universal (SUS)
- Amostragem estocástica sem reposição
  - calcula-se o *fitness* relativo para cada indivíduo (idem à roleta) e, para cada vez que um indivíduo é selecionado, subtrai-se 1 do valor esperado, até que seja 0.
- Torneio estocástico de tamanho *K*
  - seleciona-se *K* indivíduos aleatoriamente, ordena-se, e apenas o melhor (ou 2) são selecionados

Métodos de seleção X convergência

- O método de seleção, juntamente com a probabilidade de aplicação do operador de mutação são os principais fatores que levam à convergência.



Elitismo

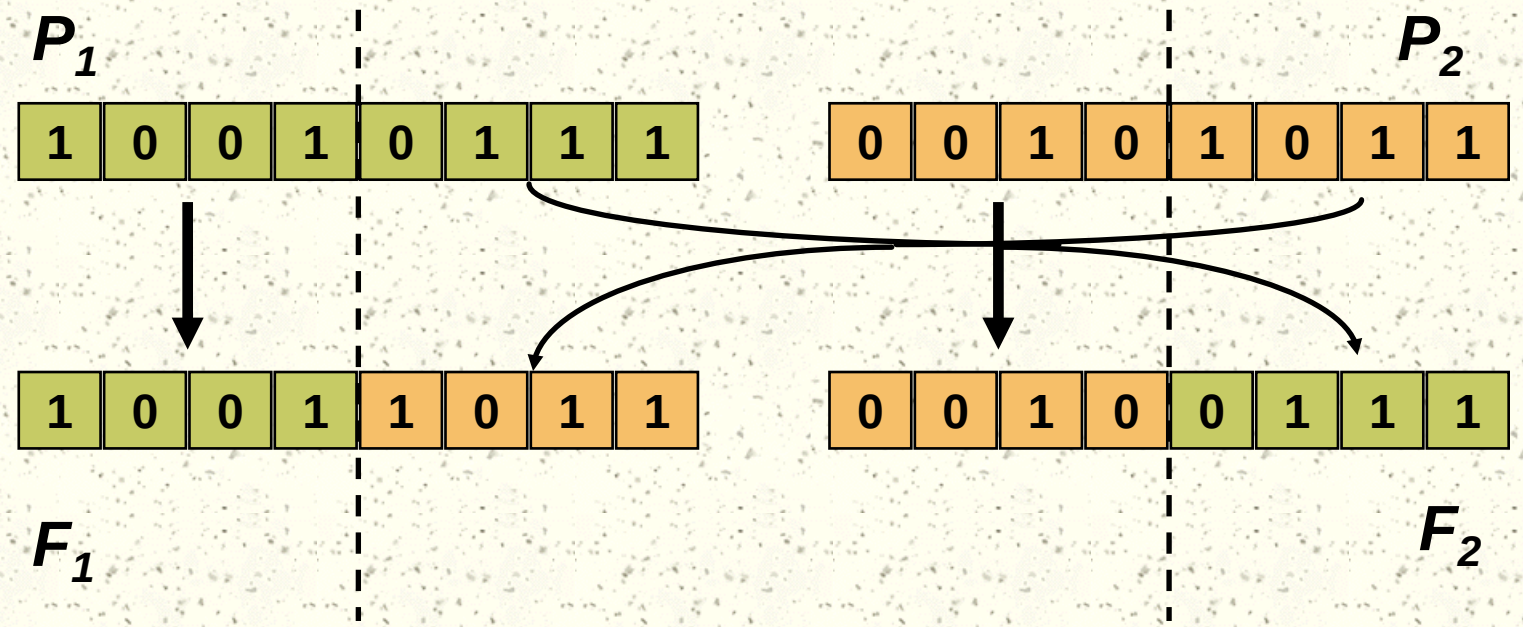
- Elitismo é a manutenção dos *k*-melhores indivíduos de uma geração para a geração seguinte.
  - É um complemento aos métodos de seleção
  - Para problemas unimodais acelera a convergência, aumentando a busca local
  - Para problemas multimodais pode tornar mais difícil a busca
  - Deve ser utilizado com cuidado pois pode induzir convergência prematura se o método de seleção for "agressivo"

SEJA $a_k(t)$ o melhor indivíduo até a geração t

SE $a_k(t) \notin P(t+1)$
ENTÃO $P(t+1) \leftarrow a_k(t)$

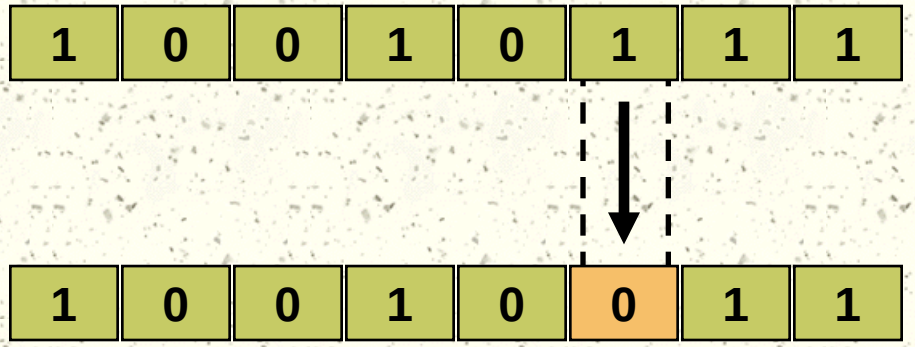
Operador de recombinação (*crossover*) para codificação binária

- Realiza busca local (*exploitation*), através de recombinação de genes de dois cromossomos-pai, gerando dois cromossomos-filho
- variações: 1-ponto, 2-pontos, uniforme...



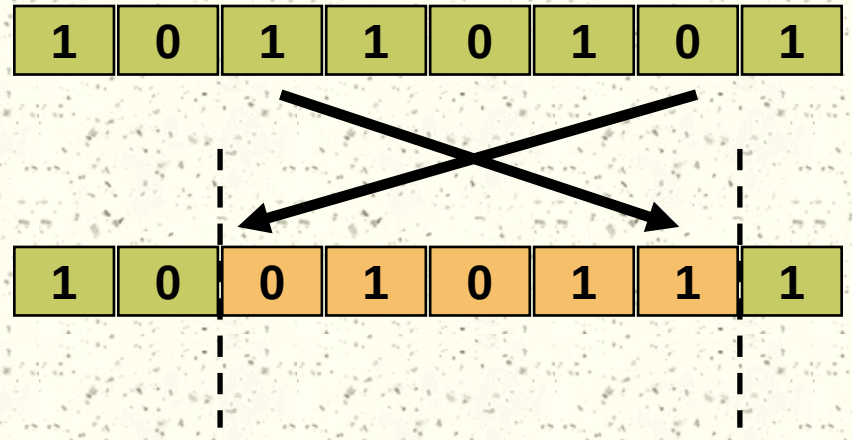
Operador de mutação para codificação binária

- Realiza busca global (*exploration*), através de uma modificação aleatória de um bit
- Objetivos:
  - Introduzir novos indivíduos na população
  - Evitar a perda irrecuperável de informação genética
- Probabilidade de mutação (p_m):
  - Muito alta: busca aleatória
  - Muito baixa: busca localizada
  - Em geral, $p_m = 1/\lambda$ por bit



Operador de inversão para codificação binária

- Tem efeito equivalente a várias mutações
- É aplicável a cromossomos de comprimento elevado.
- Raramente é utilizado.
- Pode ser útil para problemas de permutação

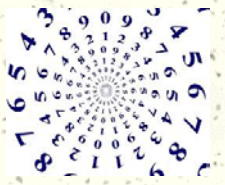


Operadores de *crossover* para codificação real

- Operadores de *crossover*:
  - *Crossover* aritmético:
    - gera os filhos como uma combinação linear dos dois vetores-pai.
  - *Crossover* simples:
    - igual à versão binária para crossover de um ponto.
  - *Crossover* heurístico:
    - Usa a função objetivo para determinar a direção da busca, produz um único descendente (ou pode não produzir nenhum).
    - Sejam X_1 e X_2 os dois vetores-pai e Y o vetor-filho, $Y=r*(X_2- X_1)+ X_2$, desde que X_2 seja uma solução melhor do que X1 (maior *fitness*), e r é um número aleatório entre 0 e 1. Este *crossover* contribui para a precisão da solução (ajuste fino).

Operadores de mutação para codificação real

- Operadores de mutação:
  - Mutação uniforme
    - um elemento x_k do vetor pai é mudado aleatoriamente no intervalo *[left(k),right(k)]*
  - Mutação limítrofe
    - igual a anterior exceto que muda para um dos dois limites, inferior ou superior
  - Mutação não-uniforme
    - o elemento selecionado será somado ou subtraído um valor aleatório que diminui à medida que avança o número de gerações

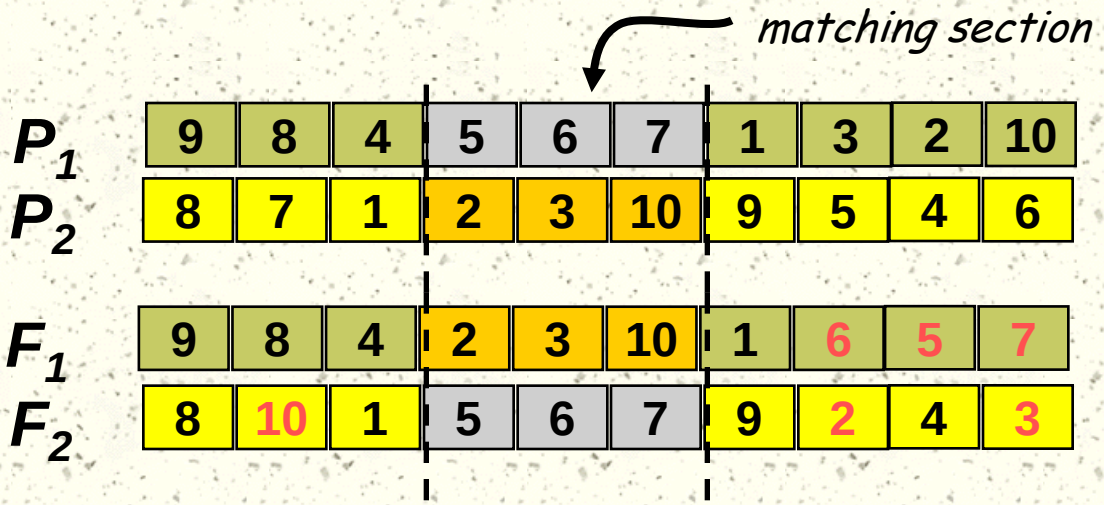


Operadores para codificação com inteiros (ordenamento)

- Utilizados em problemas combinatoriais/ permutação
- Não se pode utilizar o *crossover* tradicional pois gera soluções inválidas
- Codificação específica, em geral com números inteiros
- PMX, OX, CX, outros
- Por exemplo, para TSP depende da forma de codificação (vértices, arestas, etc)

PMX - *partially matched crossover*

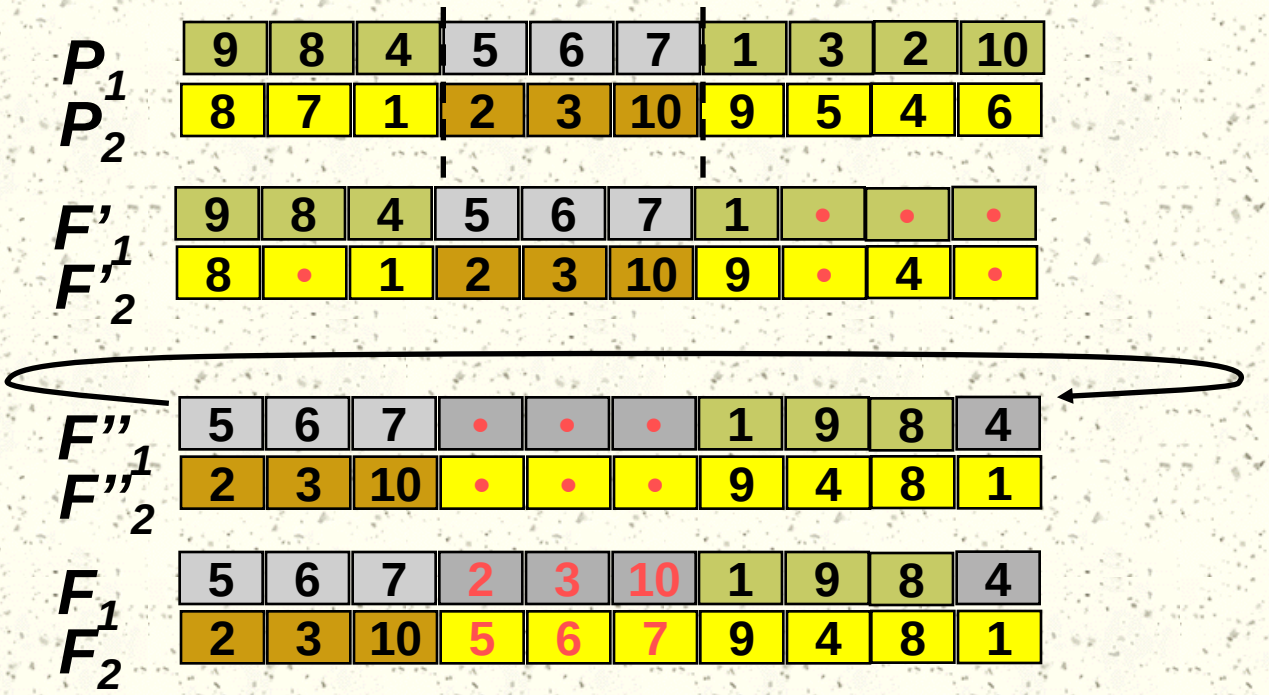
- Tende a respeitar a posição absoluta dos elementos. O algoritmo tem três passos:
  1- escolhe-se aleatoriamente dois pontos p/corte
  2- trocam-se as partes da *matching section*
  3- mapeia-se o restante dos alelos



Cada filho tem um ordenamento parcialmente determinado por cada um de seus pais

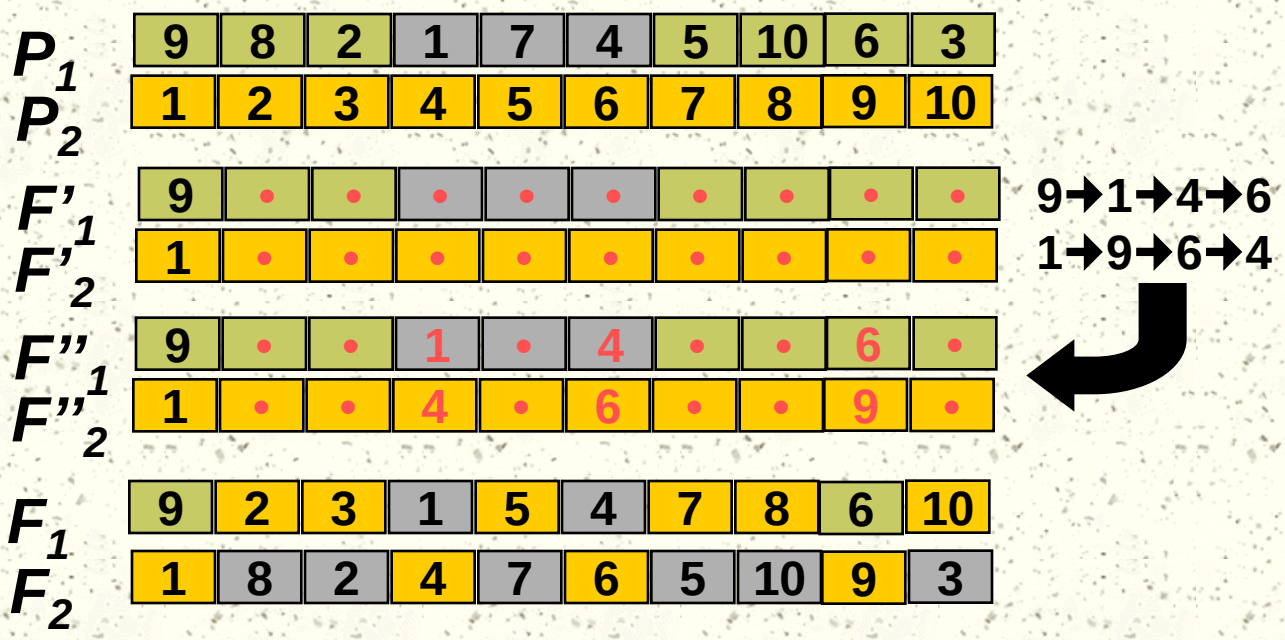
OX - *order crossover*

- Tende a respeitar a posição relativa dos elementos. O Algoritmo constroi um descendente escolhendo um *substring* de um pai e preservando a ordem relativa dos elementos no outro pai



CX - *cycle crossover*

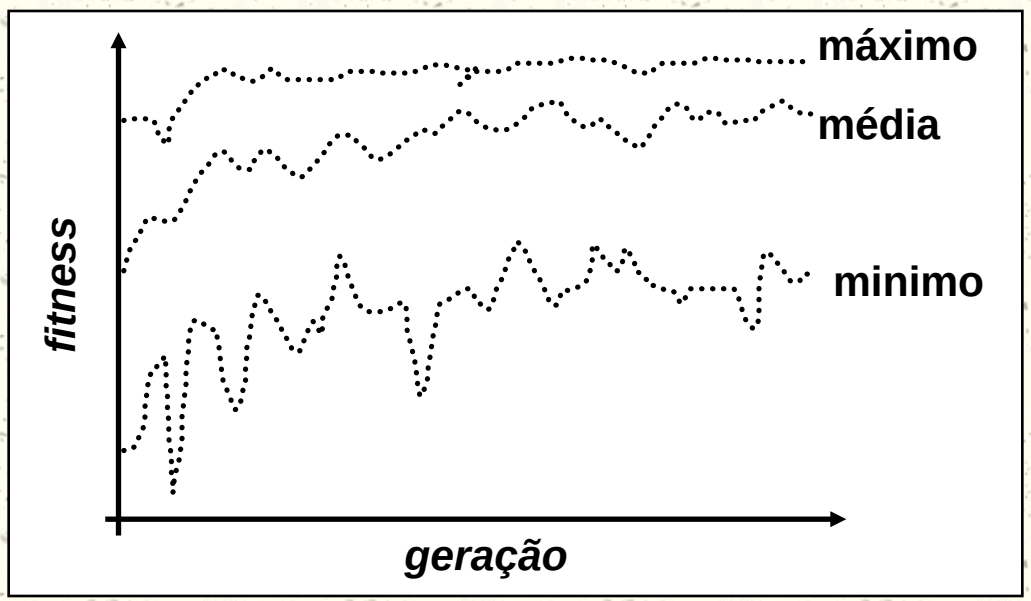
- Não usa pontos de corte
- Mantém o ponto de início para completar um ciclo
- Os filhos têm cada elemento e sua posição de um dos pais



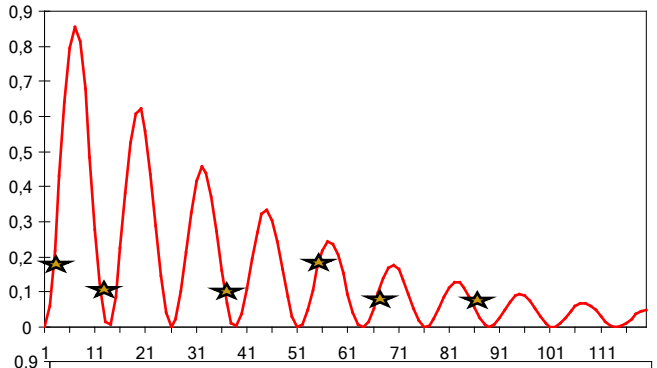
Critérios de convergência

- Critérios de convergência de DeJong:
  - Um determinado gene convergiu quando 95% da população tem o mesmo gene
  - Uma população convergiu quando todos os genes convergiram

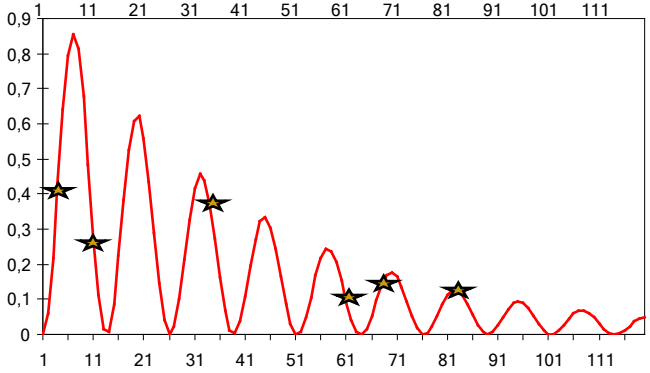
Importante: sempre observar a curva de evolução dos *fitness* !!!!!



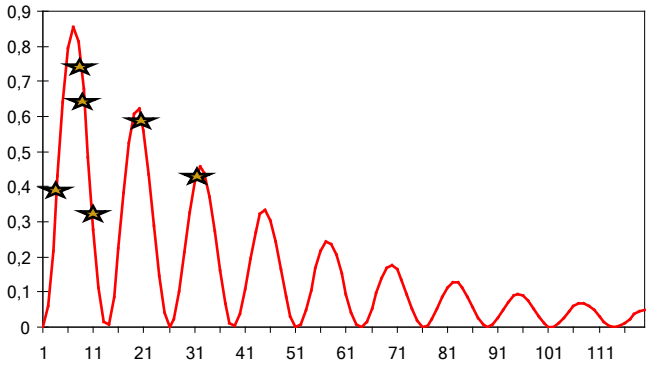
Visualização da convergência



geração=1



geração=10



geração=20

Medidas de convergência de DeJong: *On-line* e *off-line performance*

- *On-line performance* é a média de todas as avaliações de *fitness* até o momento. É uma medida de desempenho instantâneo (T=número de gerações)

$$\text{on - line performance: } x_e(\text{s}) = \frac{1}{\text{T}} \sum_{1}^{T} f_e \quad (t)$$

- *Off-line performance* é a média dos melhores indivíduos de cada geração. É uma medida de convergência

$$\text{off - line performance: } x_e^*(\text{s}) = \frac{1}{\text{T}} \sum_{1}^{T} f_e^* \quad (t)$$

$$\text{onde } f_e^*(t) = \text{melhor}\{ f_e(1), f_e(2), \ldots, f_e(t), \}$$

Definições importantes

Importante !

- Diversidade genética:
  - Medida de não-semelhança entre indivíduos de uma população.
  - Para AGs com codificação em binário a diversidade pode ser medida pela distância de Hamming entre pares de indivíduos.
- Pressão seletiva:
  - Efeito das discrepâncias de *fitness* entre indivíduos da população na preferência pela seleção.
  - Quanto maiores as diferenças, maior a pressão seletiva em benefício dos valores mais altos de *fitness*.

Efeito dos operadores na convergência

- Efeito dos operadores:
  - *crossover*: exploração local (L)
  - mutação: exploração global (G)
  - se G >> L : busca aleatória
  - se L >> G : ótimo local
- Diversidade genética:
  - No início: alta (distribuição uniforme)
  - No fim: baixa (convergência)
- Pressão seletiva:
  - relacionado ao gradiente da função de *fitness*
- Consequências:
  - convergência prematura
  - "chegada lenta"

Pressão seletiva X Diversidade Genética

- Problema:
  - nas gerações iniciais ocorre:
    - alta diversidade genética
    - alta discrepância de *fitness*
    - alta pressão seletiva
    - rápida perda de diversidade genética
    - convergência prematura
  - em gerações maduras ocorre:
    - baixa diversidade
    - baixas discrepâncias de *fitness*
    - baixa pressão seletiva
    - evolução lenta ou estagnação
- Única solução:
  - Controlar a pressão seletiva

Controle da pressão seletiva

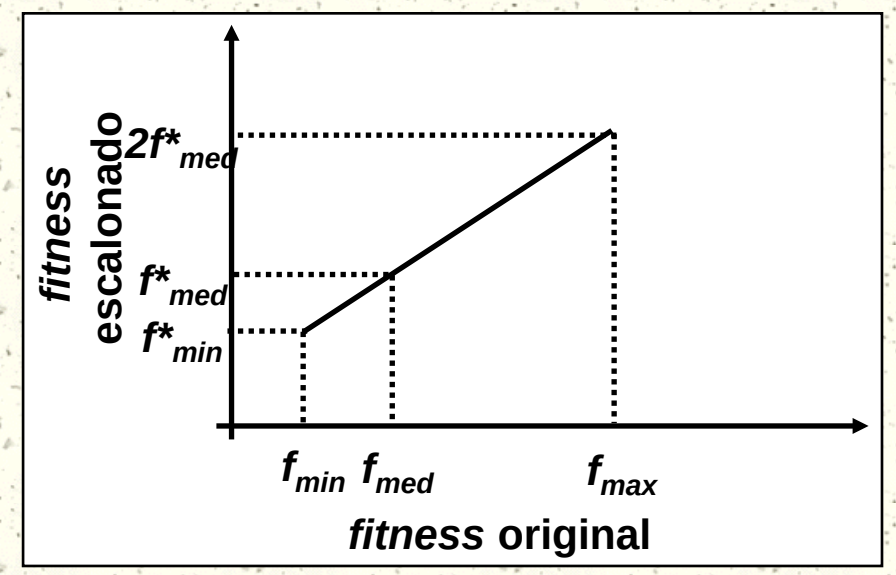
- Escalonamento linear
- Outros métodos de escalonamento
  - *ranking*, janelamento, *sigma-truncation*
- Outras estratégias
  - *Sharing*
- Métodos de seleção
  - Métodos menos "agressivos" ←

Escalonamento de *fitness*

- Escalonamento linear:
  - Objetivo: manter o *fitness* escalonado médio igual ao *fitness* não-escalonado médio
  - Promove "compressão" e "expansão" automática de escala

$$f^* = \alpha . f + \beta$$

$$f^*_{\max} = C . f_{med}$$
$$\text{onde } 1.2 \leq C \leq 2.0$$



Outros mecanismos de escalonamento

- Truncagem sigma:
  - executada antes de escalonar ou sem escalonamento

$$f^* = f - (\bar{f} - C.\sigma)$$
$$\text{onde } 1 \leq C \leq 2$$

- Janelamento:

$$f^* = f - f_{\min}$$

Generation Gap

- Foi proposto por DeJong (1975)
- É uma maneira de permitir a sobreposição ou não de duas populações consecutivas
- Modelo *generational:*
  - G=1 é o modelo tradicional sem sobreposição, onde 100% da população é substituída a cada geração.
- Modelo *steady-state*:
  - 0< G< 1 λ * G novos indivíduos são gerados para substituir parte da população. Causa uma desaceleração da velocidade de evolução.

Epistasia

- Epistasia é a influência de um gene em outro
- Não há, até o momento, uma maneira de medir epistasia.
- É praticamente inevitável para problemas reais.
- Graduação qualitativa:
  - nível 0: nenhuma interação
  - nível 1: interação moderada ou previsível
  - nível 2: interação complexa e imprevisível

- Se for muito baixa: técnicas mais simples
- Se for muito alta: AG é pouco eficiente

Problemas enganadores

- "Decepção" (*deception*) é um problema crítico em AGs
- Está intimamente relacionada com a epistasia
- *Building Blocks Hypothesis*
- A combinação de blocos construtivos bons separadamente gera uma solução de má qualidade, logo, leva a uma redução do *fitness* em vez do seu aumento.
- Única alternativa: modificar a codificação do problema

Nichos e espécies

- Inspiração na natureza onde espécies diferentes se agrupam num mesmo nicho ecológico competindo entre si pelos recursos naturais
- Em AGs é a manutenção de subpopulações estáveis com baixa competição entre as mesmas (espécies)
- Dois objetivos básicos de utilizar nichos:
  - Quando se deseja não apenas uma solução, porém um conjunto das melhores soluções
  - Permitir uma melhor exploração do espaço de busca para problemas multimodais

Fator de *crowding*

- Fator de *crowding* (fc):
  - um novo indivíduo gerado substitui o indivíduo mais semelhante a ele na população antiga, escolhido entre *fc* indivíduos amostrados aleatoriamente na população.
- Diminui a competição inter-espécies (indivíduos muito diferentes) e aumenta a competição intra-espécies
- Melhora exploração do espaço de busca através da manutenção da diversidade genética
- Útil para busca de vários sub-ótimos

Compartilhamento

- Compartilhamento (*sharing*) proposto por Goldberg (1989):
  - Os indivíduos de "uma mesma vizinhança" (mais próximos entre si) compartilham mais seus *fitness*
  - A semelhança pode ser no nível do genótipo ou do fenótipo
  - Quando há muitos indivíduos próximos, ocorre uma diminuição dos *fitness* deste grupo.
  - Este processo limita o crescimento indiscriminado de uma espécie numa região do espaço de busca

$$f_c(x_i) = \frac{f(x_i)}{\sum_{j=1}^{n} s(d(x_i, x_j))}$$

Redução de incesto

- Redução de Incesto (ou restrição de acasalamento)
  - Reduz o número de cruzamentos entre indivíduos muito semelhantes.
  - É semelhante ao *crowding* criado por DeJong
- Sofisticação do método:
  - Permitir cruzamentos somente entre elementos da mesma "família" enquanto a média de *fitness* da família for progressivo (*inbreeading*).
  - Quando isto não ocorrer, permitir o cruzamento entre "famílias" diferentes (*intermittent crossbreading*)