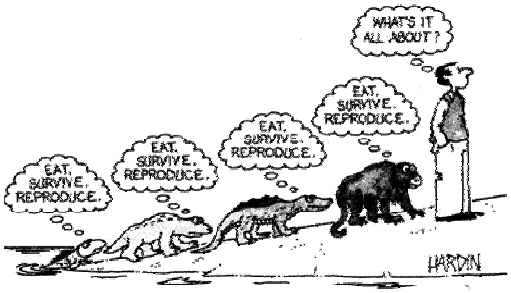
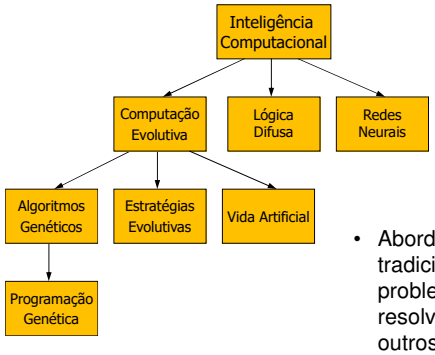


Computação Evolutiva - Conceito

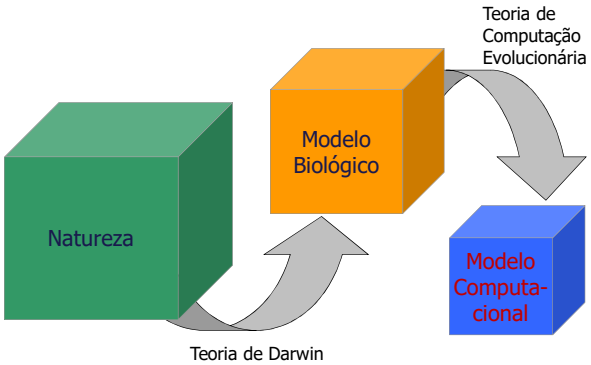
Computação Evolutiva consiste numa máquina "aprendente" otimizada, baseada nos moldes dos mecanismos de evolução biológica e seleção natural.

Computação Evolutiva

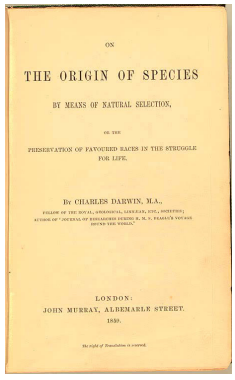


- Abordagens não tradicionais para problemas difíceis de resolver utilizando outros métodos

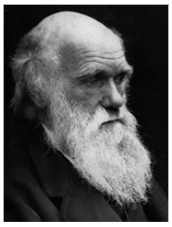
Ambientação



Teoria da Evolução - Darwin


ON
THE ORIGIN OF SPECIES
BY MEANS OF NATURAL SELECTION,
OR THE
PRESERVATION OF FAVOURED RACES IN THE STRUGGLE FOR LIFE.
By CHARLES DARWIN, M.A.,
LONDON:
JOHN MURRAY, ALBEMARLE STREET.
1859.

1859



Charles Darwin
(1809 - 1882)

O que são Algoritmos Genéticos (AGs)?

- Algoritmos Genéticos são algoritmos de busca baseados na seleção natural, onde a sobrevivência está vinculada a aptidão dos indivíduos ao ambiente.
- Seleção Natural
  - Os indivíduos mais aptos tem maior longevidade, e portanto, tem maior probabilidade de reprodução.
  - Indivíduos com maior probabilidade de reprodução tem mais descendentes, e portanto, mais chances de perpetuar seu código genético ao longo das gerações.
  - O código genético constitui a identidade de cada indivíduo e está representado no cromossoma.
- Estes princípios são aplicados na construção de algoritmos que buscam a solução ótima para um determinado problema.

Porque utilizar AGs (ou GAs)?

- AGs se aplicam a problemas com diversos parâmetros;
- AGs se aplicam a problemas não lineares e com restrições não-lineares;
- AGs se aplicam a problemas que não podem ser representados matematicamente;
- AGs se aplicam a problemas com espaços de busca enormes;
- AGs são algoritmos de busca paralela – os métodos de busca tradicionais atuam no valor da variável, varrendo o espaço de busca seguindo uma regra que determine o ponto seguinte.
  - Os AGs realizam a busca evoluindo uma população.

Algoritmos genéticos

- A IA usa sempre algumas metáforas...
  - Cérebro e sistema nervoso
    ⇒ Conexionismo
  - Linguagem + processos cognitivos
    ⇒ IA simbólica
  - Teoria da evolução
    ⇒ Computação evolutiva (algoritmos genéticos)

História da Teoria da Evolução

- 1809: Jean-Baptiste Lamarck
  - Lei do uso e do desuso
    - pelo uso e desuso de suas aptidões, a natureza força os seres a se adaptarem para sobreviverem.
  - Lei dos caracteres adquiridos.
    - Os serem mais fortes são mais capazes de "transmitir" suas aptidões às novas gerações

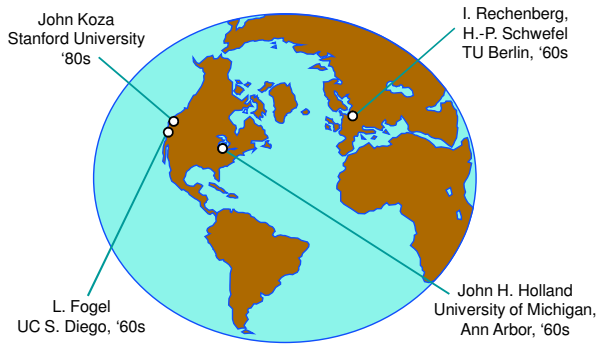
História da Teoria da Evolução

- 1859: Charles Darwin
  - Existe uma diversidade de seres devido aos contingentes da natureza (comida, clima, ...) e é pela lei da **Seleção Natural** que os seres mais adaptados ao seus ambientes sobrevivem
    - Contra lei do uso de desuso
  - Os caracteres adquiridos são herdados pelas gerações seguintes
    - O homem vem do macaco...

História da Teoria da Evolução

- 1865: Gregor Mendel
  - Formalizou a "herança de características", com a teoria do DNA (ervilhas)
- 1901: Hugo De Vries
  - Só a seleção natural não é responsável pela produção de novas (mais adaptadas) espécies. Tem de haver uma mudança genética!
  - Formalizou o processo de geração de diversidade: **Teoria da Mutação**

Precursores da Computação Evolutiva



Computação evolutiva

- 1975: Jonh Holland: Idealizou os algoritmos genéticos
  - *Adaptation in Natural & Artificial Systems* MIT Press, 1975 (2nd ed. 1992)
- Porque a evolução é uma boa metáfora?
  - Muitos problemas computacionais
    - Envolvem busca através de um grande número de possíveis soluções
    - Requerem que o programa seja adaptativo, apto a agir em um ambiente dinâmico
  - A evolução biológica
    - É uma busca maciçamente paralela em um enorme espaço de problema
    - Soluções desejadas = organismos mais adaptados

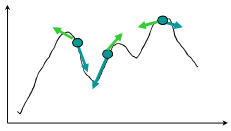
Computação Evolutiva: introdução

- Computação evolutiva
  - Método probabilista de busca para resolução de problemas (otimização) "inspirado" na teoria da evolução
  - Tem várias variantes: algoritmos genéticos, programação genética, estratégia evolutiva e programação evolutiva
- Ideia:
  - Indivíduo = solução ;
  - Provoca mudança nos indivíduos por intermédio de mutação e reprodução;
  - Seleciona indivíduos mais adaptados através de sucessivas gerações;
  - A aptidão de cada indivíduo é medida pela "função de aptidão" (*fitness function*) f(i): R → [0,1]

Algoritmos genéticos

- Aleatórios: computação evolutiva
  - indivíduo = solução

Processo adaptativo e paralelo



Aplicação: classes de problemas

- Aproximação de funções
  - Não-lineares/lineares, multi-modais
  - Discretas/contínuas
- Otimização combinatória (*NP hard*)
- Aprendizagem

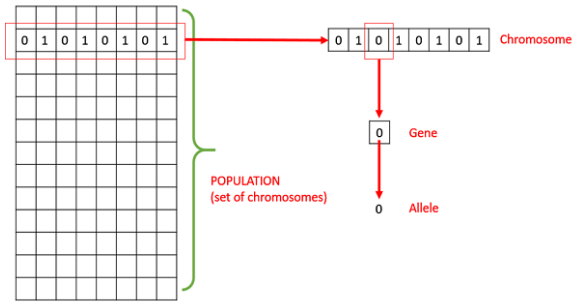
Características dos Algoritmos Genéticos

- É um algoritmo estocástico (não é determinístico);
- Trabalha com uma população de soluções simultaneamente;
- Utiliza apenas informações de custo e recompensa. Não requer nenhuma outra informação auxiliar (como por exemplo o gradiente);
- São fáceis de serem implementados em computadores;
- Adaptam-se bem a computadores paralelos;
- Funcionam com parâmetros contínuos ou discretos.

Conceitos Básicos

- AG manipula uma população de indivíduos;
- Indivíduos são possíveis soluções do problema;
- Os indivíduos são combinados (crossover) uns com os outros, produzindo filhos;
- Os filhos gerados podem ou não sofrer mutação;
- As populações evoluem através de sucessivas gerações até encontrar uma solução boa o bastante (talvez ótima).

Conceitos Básicos



Analogia com a Biologia

- **Cromossomo**: representação de uma possivel solução – individuo
- **Genes**: são as partes do cromossomo, representa uma caracteristica particular do indivíduo
- **População**: conjunto de individuos que estão sendo cogitados como solução
- **Reprodução sexual**: genes são intercambiados entre dois pais – *crossover*
- **Mutação**: os filhos são sujeitos a modificações, na qual genes elementares são modificados

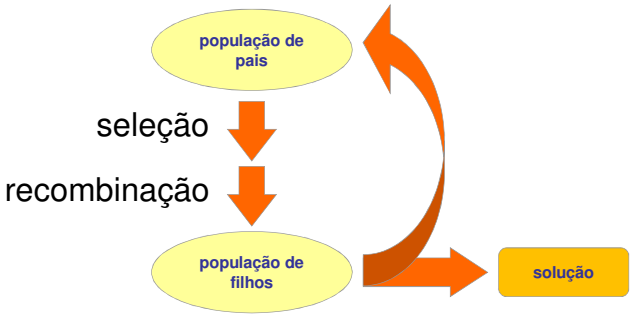
Como utilizar GAs?

- Dado que exista um problema para o qual se deseja obter a solução ótima, para utilizar GAs é necessário definir (nessa ordem):
  1. Indivíduo/Cromossoma – criar a representação das soluções do problema (inerente ao problema);
  2. Codificação/Decodificação do cromossoma – transformação do cromossoma em parâmetros do problema e vice-versa;
  3. Avaliação do indivíduo – medir o quanto a solução, representada pelos parâmetros decodificados do cromossoma do indivíduo, se aproxima do objetivo (função objetivo);
  4. Seleção dos indivíduos para as gerações seguintes;
  5. Operadores de reprodução e mutação;
  6. Inicialização da população.

Inicialização População

- Existem dois métodos principais para inicializar uma população de AG. São eles:
  - **Inicialização aleatória:** Preencher a população inicial com soluções completamente aleatórias;
  - **Inicialização heurística:** Preencher a população inicial usando uma heurística conhecida para o problema.

Visão Geral do Algoritmo Evolutivo



Algoritmo genético

```
t := 0                                  // tempo inicial
P := população inicial de indivíduos    // conjunto de
                                           soluções
Avalia aptidão de cada indivíduo de P   // função objetivo

Enquanto critérioDeParada(MaxGerações, fitness(P)) não é
   satisfeito faça:
   t := t + 1                 // incrementa tempo
   P´ := seleciona(P)         // população mais adequada
   P´´ := reproduz(P´)        // gera descendentes
   P´´´ := mutaçao(P´´)       // diversifica-os
   P := substitui(P,P´´´)     // escolhe os sobreviventes
   Avalia aptidão de P´´´
```

Questões centrais

- Como representar os indivíduos?
- Quem é a população inicial?
- Como definir a função objetivo?
- Quais são os critérios de seleção?
- Como aplicar/definir o operador de reprodução?
- Como aplicar/definir o operador de mutação?
- Como garantir a convergência e ao mesmo tempo a solução ótima?

Exemplo 1

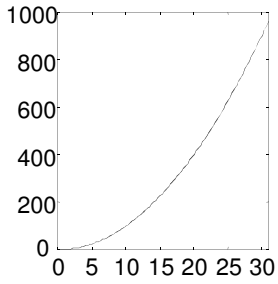
Problema: Use um AG para encontrar o ponto máximo da função:

$$f(x) = x^2$$

com x sujeito as seguintes restrições:

$$0 \le x \le 31$$

x é inteiro



Indivíduo

- Cromossomo
  - Estrutura de dados que representa uma possível solução para o problema de forma não ambígua
  - Os parâmetros do problema de otimização são representados por cadeias de valores.
  - Exemplos:
    - Vetores de reais, (2.345, 4.3454, 5.1, 3.4)
    - Cadeias de bits, (111011011)
    - Vetores de inteiros, (1,4,2,5,2,8)
    - ou outra estrutura de dados.

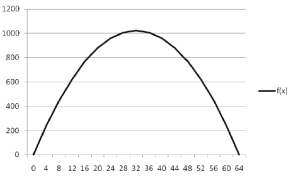
Exemplo: Representação Binária

- Encontrar o valor máximo da função f (x) = x^2, para x Ɛ [0; 63];
- Podemos representar as soluções do problema através de um cromossoma de 6 bits.
  C1: 0 0 1 0 0 1 representa x = 9
  C2: 0 0 0 1 0 0 representa x = 4

Exemplo: Representação Binária

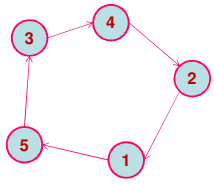
- Representação – Cadeia de bits (função)
- f(x) = 1024-(x-32)^2

i	binário	x	f(x)
1	000100	4	240
2	110111	55	495
3	011001	25	975
4	101111	47	799



Exemplo: Representação por Vetor

- Problema do Caixeiro Viajante
- C = {3, 4, 2, 1, 5}



Codificação / Decodificação do cromossoma

- A representação do cromossoma é inerente ao problema em questão, dessa forma é necessário construir a solução real do problema a partir do cromossoma;
- Exemplo: decodificação representação binária para $f(x) = x^2$:
  - O cromossomo 0 0 1 0 0 1 é decodificado para a solução x = 9 pela expressão:
    $x = 0 * 2^5 + 0 * 2^4 + 1 * 2^3 + 0 * 2^2 + 0 * 2^1 + 1 * 2^0 = 9$
- Exemplo: codificação representação binária para $f(x) = x^2$:
  - A codificado da solução x = 9 é dada pelo algoritmo:

```
for ( i =0 ; i < 6 ; i ++) {
cromossoma [ i ] = ( x >> i ) && 1;
}
```

Indivíduo

- A função de avaliação é a maneira utilizada para determinar a qualidade de um indivíduo como solução do problema em questão;
- Cada indivíduo tem um valor de aptidão (*fitness)* associado a ele;
- Aptidão (ou *fitness*)
  - Nota associada ao indivíduo que avalia quão boa é a solução por ele representada;
  - Será usada para a escolha dos indivíduos pelo módulo de seleção de pais, sendo a forma de diferenciar entre as boas e as más soluções para um problema.
- Aptidão pode ser:
  - Igual a função objetivo
  - Baseado no **ranking** do indivíduo da população

Cromossomo do Problema 1

- Cromossomos binários com 5 bits:
  - 0 = 00000
  - 31 = 11111
- Aptidão
  - Neste problema, a aptidão pode ser a própria função objetivo.
  - Exemplo:

    aptidão(00011) = *f*(3) = 9

População Inicial do Problema 1

É aleatória (mas quando possível, o conhecimento da aplicação pode ser utilizado para definir população inicial)

Pop. inicial

cromossomos	*x*	*f*(*x*)	Prob. de seleção
A_1 = 1 1 0 0 1	25	625	54,5%
A_2 = 0 1 1 1 1	15	225	19,6%
A_3 = 0 1 1 1 0	14	196	17,1%
A_4 = 0 1 0 1 0	10	100	8,7%

Probabilidade de seleção proporcional a aptidão

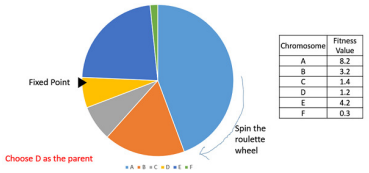
$$p_i = \frac{f(x_i)}{\sum_{k=1}^{N} f(x_k)}$$

Seleção

- Seleção
  - Tem como objetivo propagar material genético dos indivíduos mais adaptados;
  - A seleção é baseada na seleção dos indivíduos de forma que os mais aptos tem maior probabilidade de serem escolhidos para reprodução
  - Os melhores indivíduos (maior aptidão) são selecionados para gerar filhos através de *crossover* e mutação;
  - Dirige o AG para as melhores regiões do espaço de busca.
- Tipos mais comuns de seleção
  - Proporcional a aptidão (roleta russa)
  - Torneio
  - Ranking (os n mais adaptados)

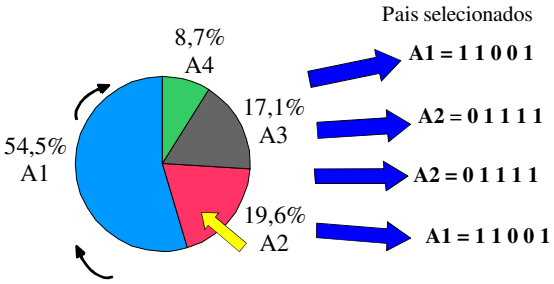
Seleção proporcional a aptidão (Roleta)

- Método simples e muito adotado: método da roleta viciada.
  - Criamos uma roleta (virtual) na qual cada cromossomo recebe um pedaço proporcional à sua avaliação (a soma dos pedaços não pode superar 100%);
  - Rodamos a roleta (isto é, sorteamos um número);
  - Selecionado será o indivíduo sobre o qual ela parar.



Chromosome	Fitness Value
A	8.2
B	3.2
C	1.4
D	1.2
E	4.2
F	0.3

Seleção proporcional a aptidão (Roleta)



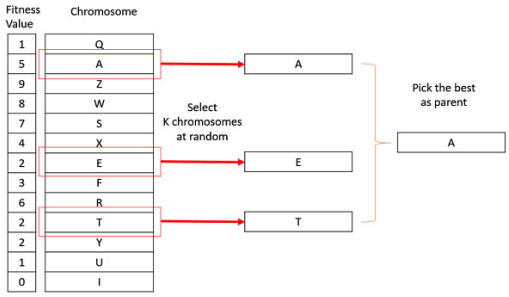
Problema: converge muito rápido por causa da variância que é pequena

Roleta

```
Inicio
   T = soma dos valores de aptidão de todos os
   indivíduos da população
   Repita N vezes para selecionar n indivíduos
      r = valor aleatório entre 0 e T
      Percorra sequencialmente os indivíduos da
      população, acumulando em S o valor de aptidão
      dos indivíduos já percorridos
      Se S >= r então
         Selecione o indivíduo corrente
      Fim se
   Fim Repita
Fim
```

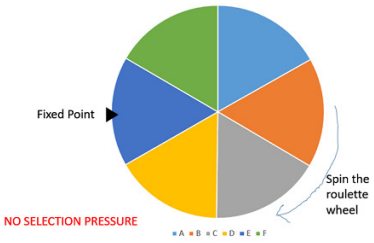
Seleção por Torneio

- Torneio: escolhe-se *n* (tipicamente 2) indivíduos aleatoriamente da população e o melhor é selecionado.



Seleção por Ranking

- Ranking: seleciona-se os *n* indivíduos mais adaptados



Chromosome	Fitness Value
A	8.1
B	8.0
C	8.05
D	7.95
E	8.02
F	7.99

Reprodução - *Crossover*

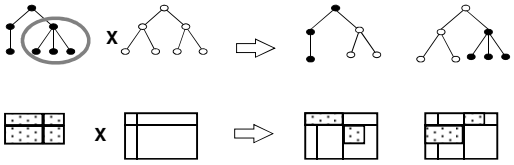
- Função:
  - combinar e/ou perpetuar material genético dos indivíduos mais adaptados;
  - Cria novos indivíduos misturando características de dois ou mais indivíduos pais (*crossover*) – variação.
- Em termos de busca:
  - Principais mecanismos de busca do AG;
  - Permite explorar áreas desconhecidas do espaço de busca.

Crossover ou Recombinação

- Cruzamento uniforme: os filhos são formados a partir dos bits dos pais (sorteado)
- Cruzamento em um ponto
  - Pai 1: 1010101011 | 0101010111
  - Pai 2: 0000100101 | 0101110010
  - Filho1: 10101010110101110010
  - Filho2: 00001001010101010111
- Cruzamento multi-ponto
  - Pai 1: 101010 | 101101 | 01010111
  - Pai 2: 000010 | 010101 | 01110010
  - Filho1: 000010 | 101101 | 01110010
  - Filho2: 101010 | 010101 | 01010111

Crossover

- Os pontos de corte dos cruzamentos em um ponto ou multi-ponto podem ser estáticos ou escolhidos aleatoriamente
- Quanto mais estruturada for a representação do cromossomo, mais difícil fica de se definir o cruzamento



Substituição

- Objetivo:
  - garantir uma convergência adequada
- Tipos:
  - simples: a nova geração SUBSTITUI a antiga
  - elitista ou *steady-state*: a nova geração se MISTURA com a antiga
- Critérios de substituição no caso elitista:
  - os piores;
  - os mais semelhantes
    - para evitar convergência prematura
  - os melhores;
  - os pais;
  - aleatoriamente, ...

Operadores Genéticos – Mutação

- Os novos indivíduos gerados ainda podem sofrer mutações alterando pontualmente as características herdadas na operação de crossover;
- Exemplo: Mutação sobre um novo indivíduo D2

D2 0 0 0 0 0 1	x = 1	f (x) = 1
D2 0 1 0 0 0 1	x = 19	f (x) = 361

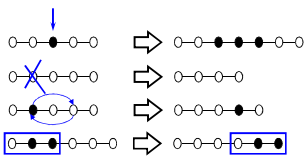
- O bit afetado pela mutação é selecionado aleatoriamente e a mutação ocorre com probabilidade *pm < 1%* (probabilidade de mutação);
- A operação de mutação explora o espaço de soluções em busca de características ausentes na população.

Mutação: Cuidados

- Valor da probabilidade deve ser baixo.
  - Se ele for muito alto, o algoritmo genético se parecerá muito com uma técnica chamada "random walk".
- Alguns textos preferem que o operador de mutação não aja de forma aleatória, mas sim, alterando o valor do gene para outro valor válido do nosso alfabeto genético.
  - Corresponde em multiplicar a probabilidade do operador de mutação por n/(n-1), onde n é a cardinalidade do alfabeto genético.

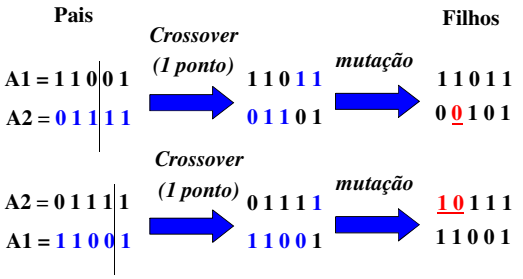
Mutação

- Objetivo:
  - gerar diversidade (p/ escapar de ótimos locais)
- Tipos:
  - Gerativa
  - Destrutiva
  - Swap
  - Swap de seqüência



- Obs: Existe uma "taxa de mutação" (ex. % da população selecionada) que diminui com o tempo para garantir convergência

Crossover e mutação do Problema 1



A primeira geração do Problema 1

- Substituição simples

cromossomos		x	$f(x)$	prob. de seleção
1	1 1 0 1 1	27	729	29,1%
2	1 1 0 0 1	25	625	24,9%
3	1 1 0 0 1	25	625	24,9%
4	1 0 1 1 1	23	529	21,1%

As demais gerações do Problema 1

Segunda Geração

		x	$f(x)$
1	1 1 0 1 1	27	729
2	1 1 0 0 0	24	576
3	1 0 1 1 1	23	529
4	1 0 1 0 1	21	441

Terceira Geração

		x	$f(x)$
1	1 1 0 1 1	27	729
2	1 0 1 1 1	23	529
3	0 1 1 1 1	15	225
4	0 0 1 1 1	7	49

As demais gerações do Problema 1

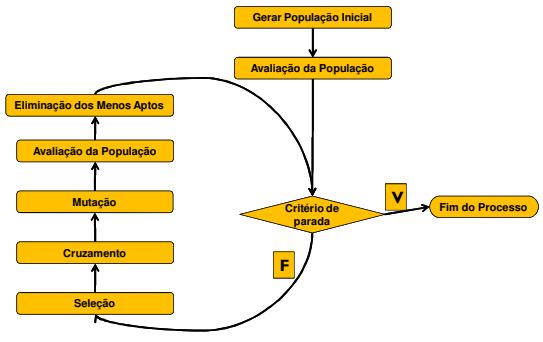
Quarta Geração

		x	$f(x)$
1	1 1 1 1 1	31	961
2	1 1 0 1 1	27	729
3	1 0 1 1 1	23	529
4	1 0 1 1 1	23	529

Quinta Geração

		x	$f(x)$
1	1 1 1 1 1	31	961
2	1 1 1 1 1	31	961
3	1 1 1 1 1	31	961
4	1 0 1 1 1	23	529

Algoritmo Genético



Problema 2

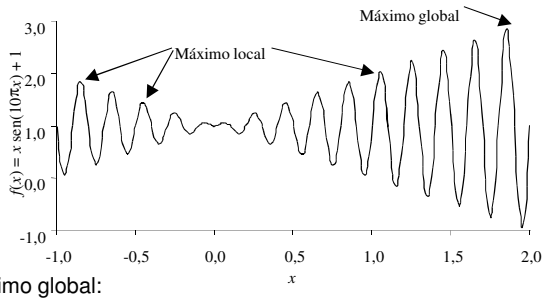
Achar o máximo da função utilizando Algoritmos Genéticos

$$f(x) = x \operatorname{seno}(10\pi x) + 1{,}0$$

Restrita ao intervalo:

$$-1{,}0 \le x \le 2{,}0$$

Problema 2



Máximo global:
x = 1,85055
f(x) = 2,85027

Problema 2

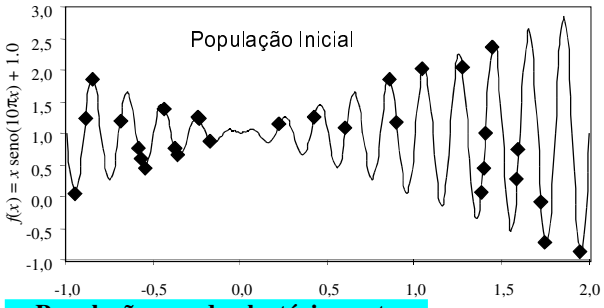
- Função multimodal com vários pontos de máximo.
- É um problema de otimização global (encontrar o máximo global)
- Não pode ser resolvido pela grande maioria dos métodos de otimização convencional.
- Há muitos métodos de otimização local, mas para otimização global são poucos.

O Cromossomo Problema 2

- Representar o único parâmetro deste problema (a variável *x*) na forma de um cromossomo:
  - Quantos bits deverá ter o cromossomo?
  - Quanto mais bits melhor precisão numérica.
  - Longos cromossomos são difíceis de manipular.
- Para cada decimal é necessário 3,3 bits
  - Cromossomo com 22 bits

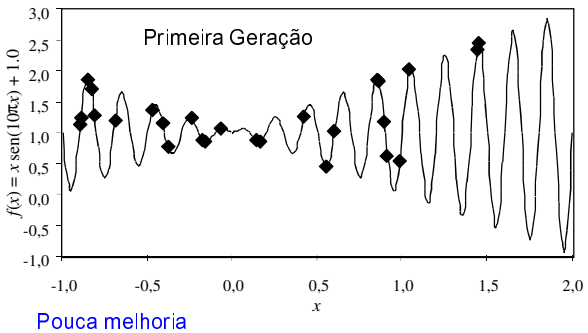
1000101110110101000111

As Gerações do Problema 2



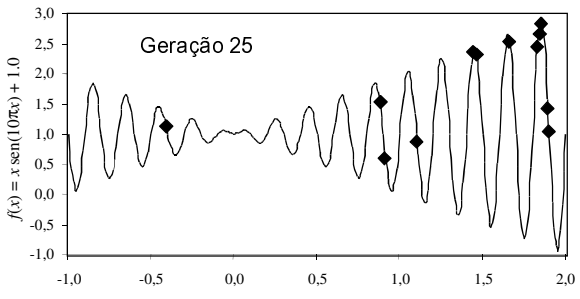
População gerada aleatóriamente

As Gerações do Problema 2



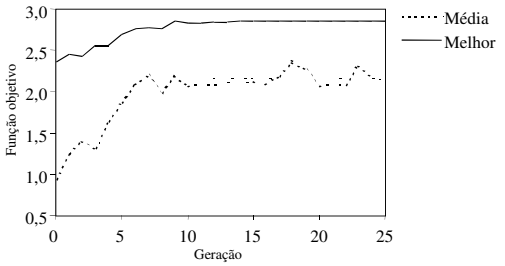
Pouca melhoria

As Gerações do Problema 2



A maioria dos indivíduos encontraram o máximo global

As Gerações do Problema 2

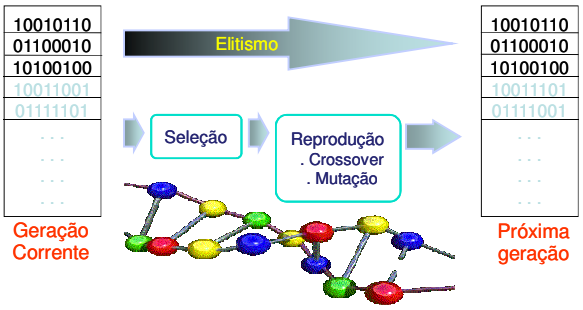


Na geração 15 o AG já encontrou o ponto máximo

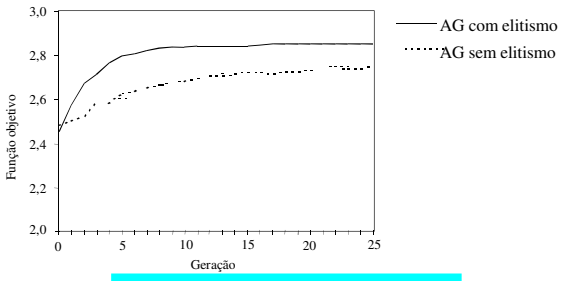
Elitismo

- A substituição simples da geração antiga, pela nova, podem destruir a melhor indivíduo;
- Por que perder a melhor solução encontrada?
- Elitismo transfere cópias dos melhores indivíduos para a geração seguinte

Elitismo



Elitismo no Problema 2



AG com elitismo é melhor ?

Critérios de Parada

- Número de gerações
- Encontrou a solução (quando esta é conhecida)
- Perda de diversidade (estagnação)
- Convergência
  - nas últimas *k* gerações não houve melhora na aptidão

- Algoritmos Genéticos - Conceitos básicos

- Parâmetros
  - Tamanho da população;
  - Taxa de cruzamento;
  - Taxa de mutação;
  - Taxa de substituição.

- Algoritmos Genéticos - Conceitos básicos

- Tamanho da população
- Se pequeno
  - Executa rápido;
  - Baixa qualidade.
- Se grande
  - Boa qualidade;
  - Custo computacional.

- Algoritmos Genéticos - Conceitos básicos

- Taxa de cruzamento
- Se pequeno
  - Convergência demorada.
- Se grande
  - Perda de material genético.

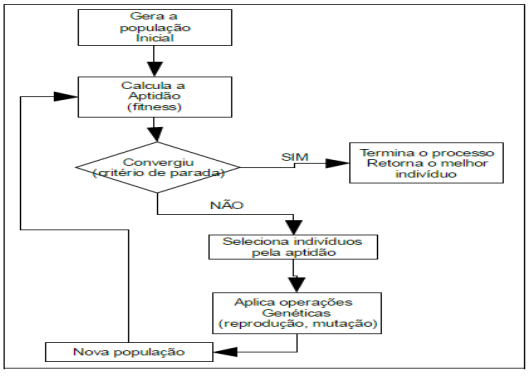
- Algoritmos Genéticos - Conceitos básicos

- Taxa de mutação
- Previne a permanência em espaço de busca limitado
  - Máximos locais.
- Se muito elevado
  - Busca aleatória (ruim).

- Algoritmos Genéticos - Conceitos básicos

- Taxa de substituição
- Quantidade de indivíduos a ser descartada
  - Bons sobrevivem;
  - Menos aptos são excluídos;
  - Material genético desconsiderado.

Esquema de execução



Exemplos de aplicações

- Roteamento de Telecomunicações
- Avaliação de Crédito e Análise de Risco
- Particionamento de circuitos
- Escalonamento
- Jogos

- **Taxa de Cruzamento**
  A taxa de cruzamento deve em geral ser alta, cerca de **80%-95%**.
  - Entretanto, alguns resultados mostram que para alguns tipos de problemas, uma taxa de cruzamento de cerca de 60% é o melhor.
- **Taxa de Mutação**
  Por outro lado, a taxa de mutação deve ser muito baixa.
  - As melhores taxas parecem estar na faixa de **0.5%-1%**.
- **Tamanho da População**
  Um bom tamanho para a população é cerca de **20-30;**
  - Entretanto às vezes tamanhos de 50-100 são relatados como os melhores.
  - Alguns autores também mostram que o melhor tamanho da população depende do **tamanho da série codificada** (cromossomas).
  - Isto significa que se você tem cromossomas com 32 bits, a população deve ser maior do que se você tivesse cromossomos com 16 bits.

Parâmetros Genéticos (1/2)

- É importante também, analisar de que maneira alguns parâmetros influem no comportamento dos Algoritmos Genéticos, para que se possa estabelecê-los conforme as necessidades do problema e dos recursos disponíveis.
- **Tamanho da População:** o tamanho da população afeta o desempenho global e a eficiência dos AGs.
  - 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.

Parâmetros Genéticos (2/2)

- **Taxa de Cruzamento:** quanto maior for esta taxa, mais rapidamente novas estruturas serão introduzidas na população.
  - Entretanto, isto pode gerar um efeito indesejado pois a maior parte da população será substituída, ocorrendo 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 tornar-se muito lento para oferecer uma resposta aceitável.
- **Taxa de Mutação:** mesmo uma baixa taxa de mutação previne que uma dada posição fique estagnada em um valor, além de possibilitar que se chegue em qualquer ponto do espaço de busca.
  - Porém deve tomar cuidado, uma vez que uma taxa de mutação muito alta pode tornar a busca essencialmente aleatória.
- **Intervalo de Geração:** controla a porcentagem da população que será substituída durante a próxima geração.

Balanço

- Vantagens
  - Simples (várias representações, 1 algoritmo)
  - Vasto campo de aplicações
  - Ainda custa caro mas pode ser paralelizado facilmente
- Desvantagens
  - Pode ser caro computacionalmente
  - Como o método é basicamente numérico nem sempre é fácil introduzir conhecimento do domínio

Vantagens dos AGs:

- Não requer qualquer informação derivada (que pode não estar disponível para muitos problemas do mundo real);
- É mais rápida e mais eficiente em comparação com os métodos tradicionais;
- Elevada capacidade de paralelismo;
- Otimiza tanto as funções contínuas quanto as discretas e também problemas com vários objetivos;
- Fornece uma lista de "boas" soluções e não apenas uma única solução;
- Sempre recebe uma resposta para o problema, que fica melhor com o tempo.
- Útil quando o espaço e estados é muito grande e há um grande número de parâmetros envolvidos.

Limitações dos AGs

- Como qualquer técnica, os AGs também sofrem de algumas limitações, tais como:
  - A técnica de AG não é adequada a todo o tipo de problemas, em especial os problemas que são simples e para o qual a informação está disponível diretamente;
  - O valor de fitness (ou aptidão) é calculado repetidamente. o que pode ser computacionalmente caro para alguns problemas;
  - Sendo estocástica (não é determinística), não há garantias sobre a otimização, ou a qualidade da solução;
  - Se não for implementado corretamente, o AG pode não convergir para a solução ótima.

Links

- Course on Genetic Algorithms
  - http://gal4.ge.uiuc.edu/ge493/ge493.top.html
- Intro to GAs (slides)
  - http://lancet.mit.edu/~mbwall/presentations/IntroToGAs/
- GA faq
  - http://www.cis.ohio-state.edu/hypertext/faq/usenet/ai-faq/genetic/top.html
- Links on Genetic Algorithms
  - http://www.ics.hawaii.edu/~sugihara/research/link-dga.html