Inteligência Artificial

Busca local e busca baseada em população

Debora Medeiros, Charles Ferreira

October 19, 2021

Visão geral

Algoritmos de busca

Até agora: exploração sistemática do espaço de busca

- Mantendo um ou mais caminhos na memória
- Registrando alternativas exploradas e não exploradas

Há problemas em que o caminho é irrelevante e o estado final em si é a solução

- projeto de circuitos
- layout de instalações
- 8 rainhas, basta mostrar o tabuleiro final

Meta heurística

Método heurístico para resolver de forma genérica problemas de busca e otimização

- Não necessariamente guardam caminho à solução
  - Usando então pouca memória
- Podem encontrar soluções razoáveis para problemas grandes
  - Até mesmo infinitos

Problemas de otimização

- Solução candidata: vetor de variáveis
- Função objetivo: f : solução candidata $\rightarrow \mathbb{R}$
- Objetivo: achar a solução que maximize ou minimize a função objetivo

Espaço de busca

Algoritmo completo sempre encontra uma solução, caso ela exista
Algoritmo ótimo sempre encontra mínimo/máximo global

Busca com melhoria iterativa

A ideia é começar com uma solução inicial e melhorá-la iterativamente

As soluções candidatas podem ser representados sobre uma superfície

- A **altura** de qualquer ponto na superfície corresponde à função de **avaliação** da solução naquele ponto

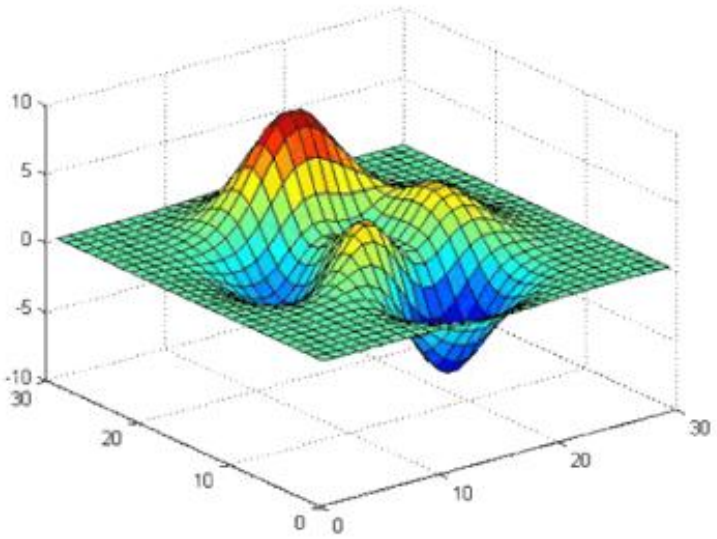
O algoritmo se "move" pela superfície em busca de pontos mais altos/baixos

- Ponto mais alto/baixo (máximo/mínimo global) corresponde à solução ótima
  - Solução onde a função de avaliação atinge seu valor máximo/mínimo

Espaço de soluções candidatas

Espaços mais complexos podem ter muitas dimensões

Exemplo: Soluções são codificadas pelas coordenadas X-Y, a qualidade da solução é a coordenada Z



Exemplo: Caixeiro Viajante

Busca por árvore

- Começa na cidade de origem
- Visita a próxima cidade até encontrar uma viagem ótima

Método de melhoria iterativa

- Começa com uma viagem aleatória
- Troca cidades até encontrar uma viagem ótima

Busca com melhoria iterativa

Esses algoritmos guardam apenas o estado atual, e não veem além dos vizinhos imediatos do estado

Contudo, muitas vezes são os melhores métodos para tratar problemas reais muito complexos

Duas classes de algoritmos

Hill-Climbing: Subida de Encosta

- Só faz modificações que melhoram o estado atual

Baseada em população

- Evolui uma população de soluções por meio de modificações de seus integrantes

Busca subida de encosta

Hill climbing

Se move de forma contínua em valor crescente

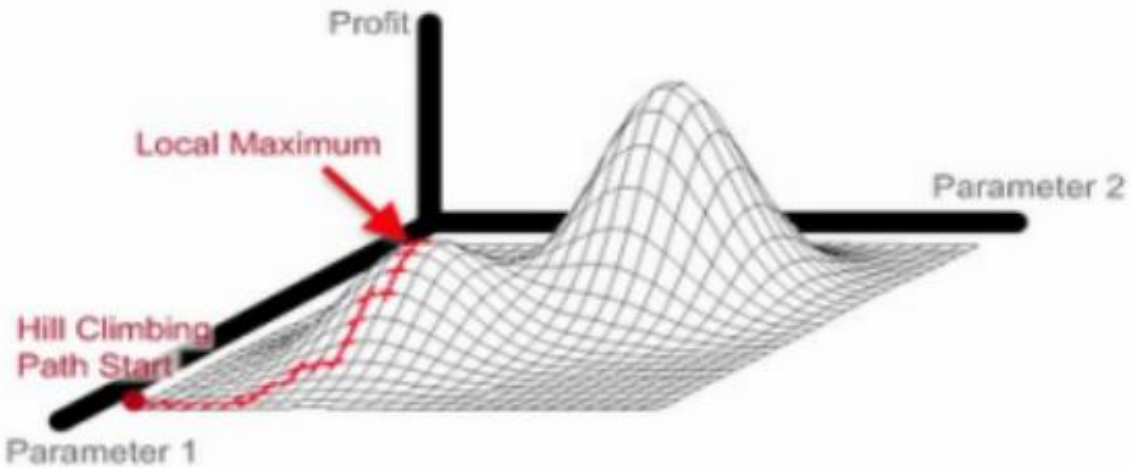
- Encosta acima

Termina quando alcança um pico

- Em que nenhum vizinho tem valor mais alto

Examina vizinhos imediatos

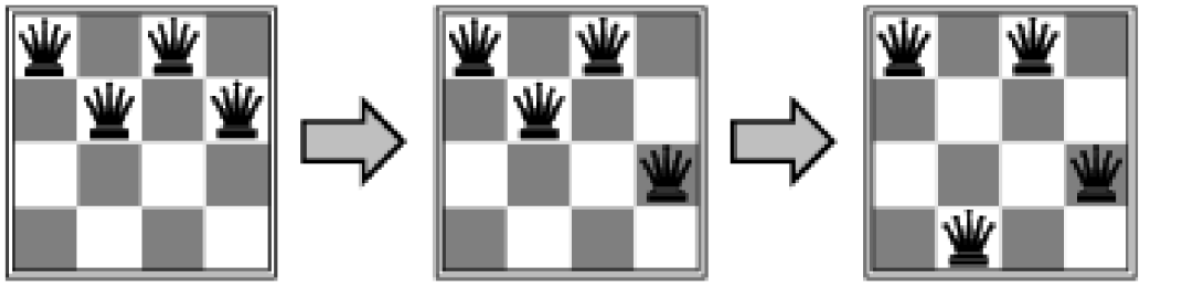
- Não precisa manter árvore de busca
- Guarda só estado corrente e tenta melhorá-lo



Ex.: n-rainhas

Busca subida de encosta

Começa por um estado e vai tentando melhorá-lo sucessivamente



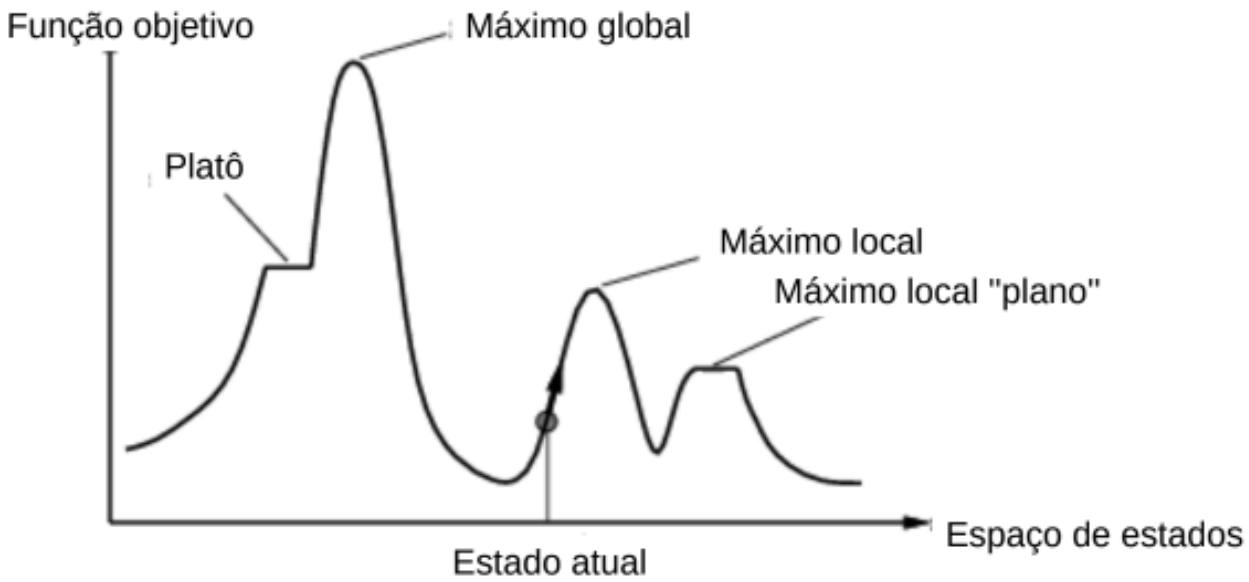
Busca subida de encosta

Chamada de busca gulosa local

Captura uma boa solução vizinha

Porém, pode ficar paralisada

- Em máximos/mínimos locais
- Em platôs



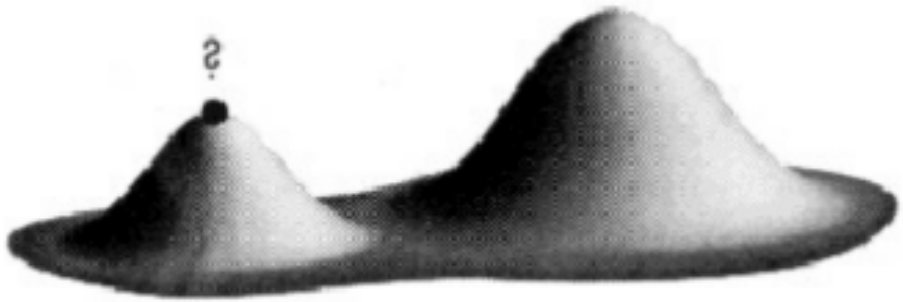
Busca subida de encosta

Máximos Locais:

Em contraste com máximos globais, são picos mais baixos do que o pico mais alto no espaço de soluções (solução ótima)

A função de avaliação leva a um valor máximo para o caminho sendo percorrido: essa função utiliza informação local

porém, o nó final está em outro mais alto



Busca subida de encosta

Platôs:

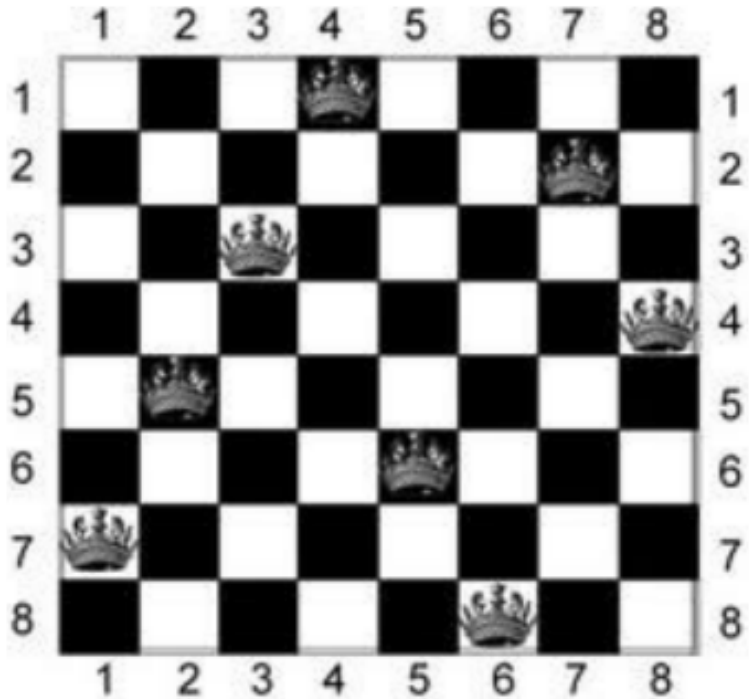
uma região do espaços de soluções onde a função de avaliação dá o mesmo resultado

Busca subida de encosta

Problema das oito rainhas: formulação de soluções completas

Cada estado tem 8 rainhas no tabuleiro, uma por coluna
Novas soluções candidatas são gerados movendo uma rainha para outro quadrado na mesma coluna

- Cada estado tem $8 * 7$ sucessores



Busca subida de encosta

Problema das oito rainhas: Função de avaliação h

- h = número de pares de rainhas se atacando
  - Neste caso, quanto menor, melhor
- Mínimo global é 0
- Escolhe melhor sucessor corrente
  - Se houver vários, escolhe aleatoriamente um deles

h = 17 → solução atual
h = 12 → melhor solução vizinha

Busca subida de encosta

Minimo local com h = 1

Todo sucessor tem custo mais alto

Alcançada em 5 passos a partir da solução inicial

Busca subida de encosta

Problema das 8 rainhas

Estados iniciais aleatórios

86% das vezes busca fica paralisada

resolve apenas 14% das instâncias do problema

- Mas é rápida

4 passos em média quando tem sucesso
3 passos em média quando fica paralisada
Em espaço que tem cerca de 17 milhões de soluções

Busca subida de encosta

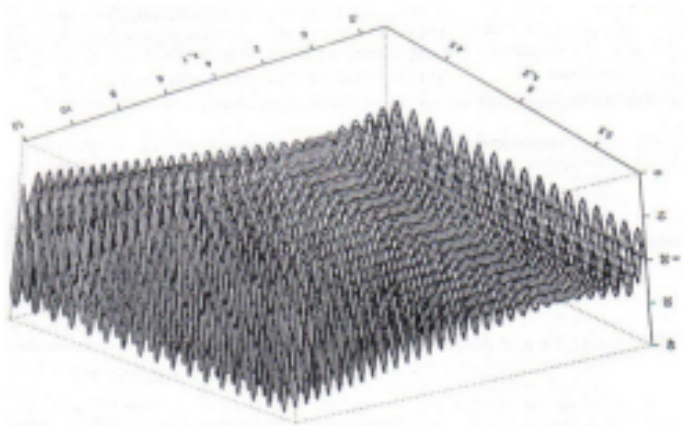
- Muda de estado em platôs
  - Quando estados têm avaliações iguais
  - Colocando um limite de vezes
- Ex. Problema das 8 rainhas
  - Com soluções iniciais aleatórias, usando essa estratégia
    - Passa a resolver 94% das instâncias do problema
      Mas demora mais
      - 21 passos em média quando tem sucesso
      - 64 passos em média quando falha

Busca subida de encosta

Sucesso depende da topologia do espaço de soluções

Se houver poucos máximos locais e platôs

- Subida de encosta com reinício aleatório encontrará boa solução com rapidez
- Mesmo para problemas mais complexos, pode encontrar máximo local razoavelmente bom
  - Com poucos reinícios



Busca subida de encosta

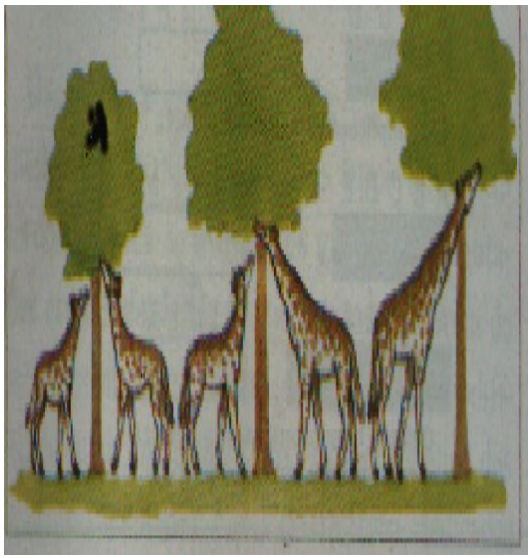
Exercício

Aplique a busca de subida de encosta ao seguinte tabuleiro

Algoritmos Genéticos

Engloba métodos e técnicas computacionais inspirados:

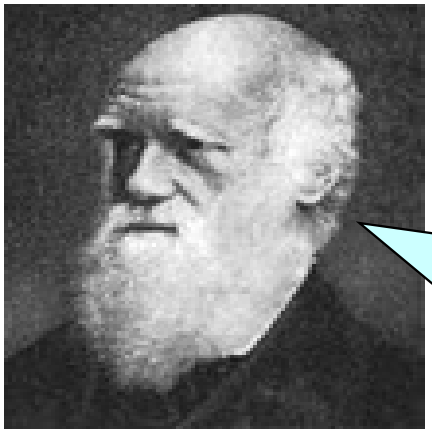
- na **teoria da evolução das espécies**, de seleção natural (Darwin)
- na **Genética** iniciada por Mendel



Bases da evolução:

- **diversidade** é gerada por cruzamento e mutações
- os seres **mais adaptados** aos seus ambientes sobrevivem
- as **características genéticas** de tais seres são **herdadas** pelas próximas gerações

Algoritmos Genéticos



1859 - Charles Darwin publica o livro “A Origem das Espécies”

As espécies evoluem pelo princípio da seleção natural e sobrevivência do mais apto

Charles Darwin

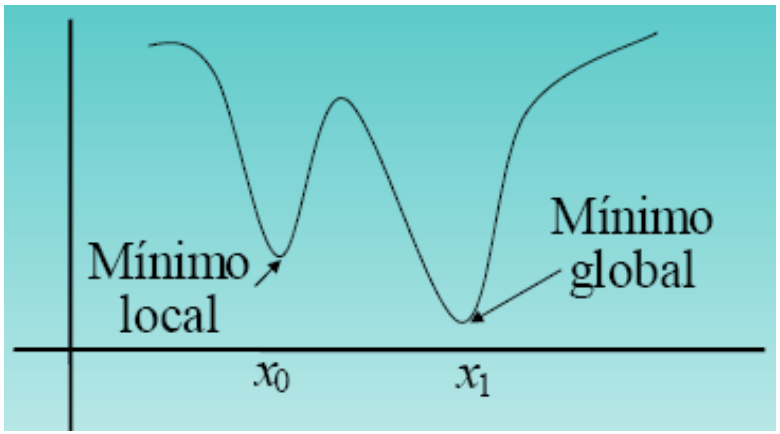
1865 - Gregor Mendel apresenta experimentos do cruzamento genético de ervilhas

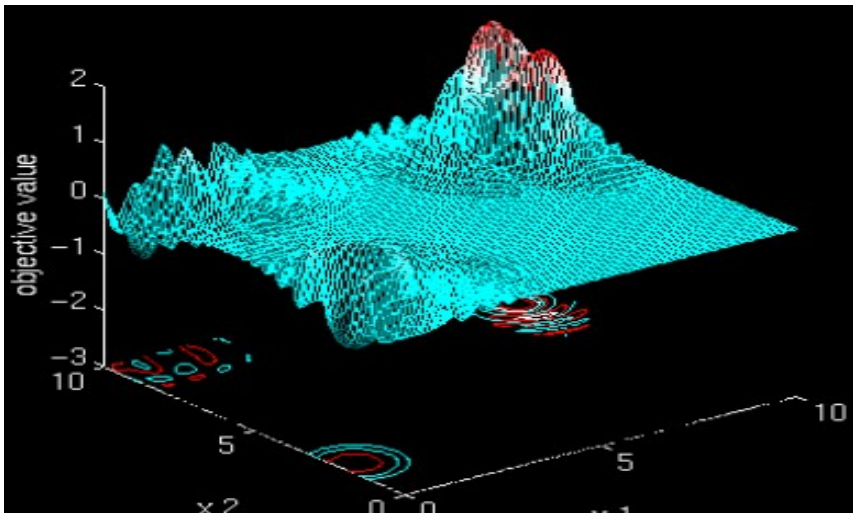
Pai da Genética

Gregor Mendel

Algoritmos Genéticos

Nos anos 1960 John Holland e seus alunos propuseram a construção de um algoritmo de busca e otimização: os algoritmos genéticos



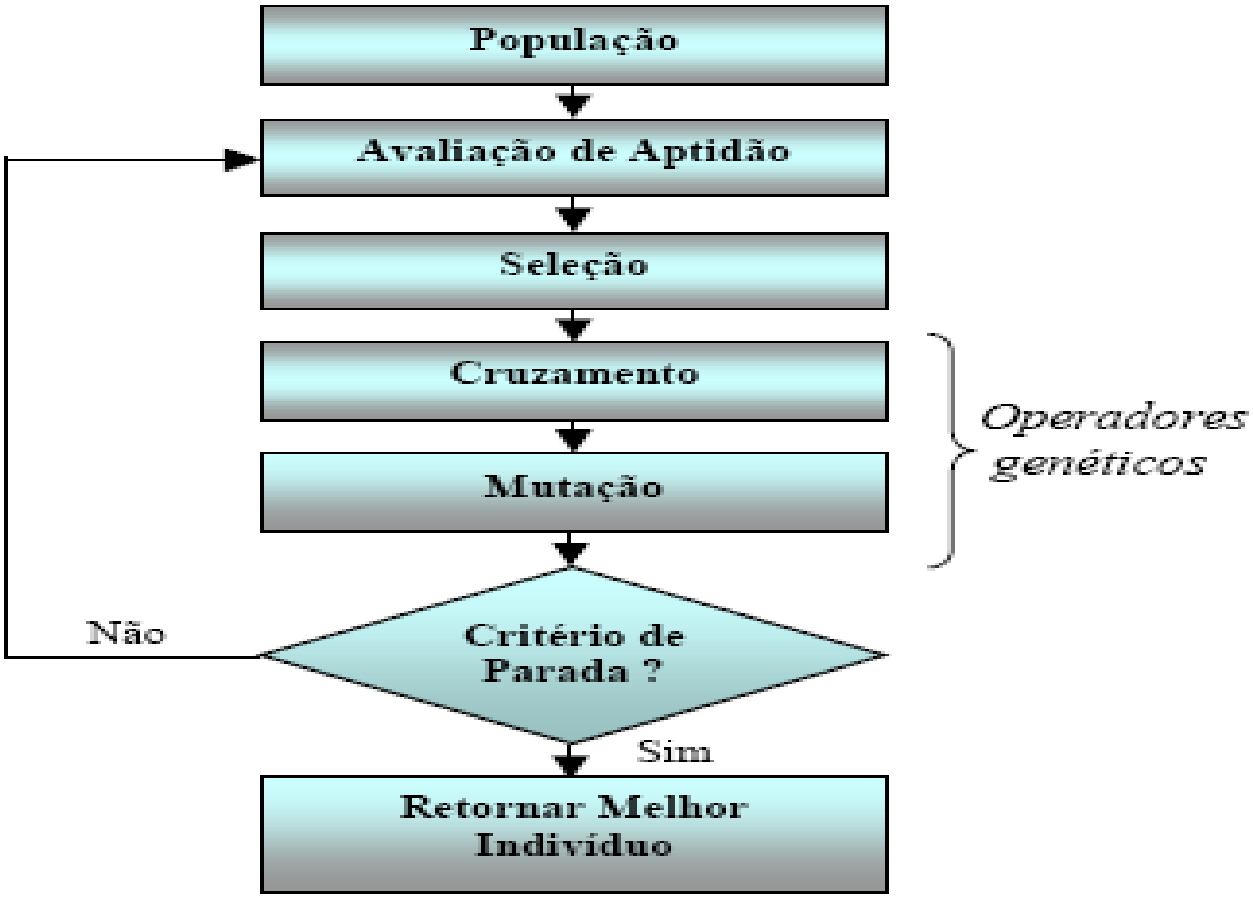


Algoritmos Genéticos

Os algoritmos genéticos usam como base, e procuram **combinar**:

- A teoria da evolução das espécies - a sobrevivência das estruturas/soluções mais adaptadas a um ambiente/problema
- Estruturas genéticas - utilizam conceitos de **hereditariedade** e **variabilidade genética** para **troca de informações** entre as estruturas visando a melhoria das mesmas

Algoritmos Genéticos

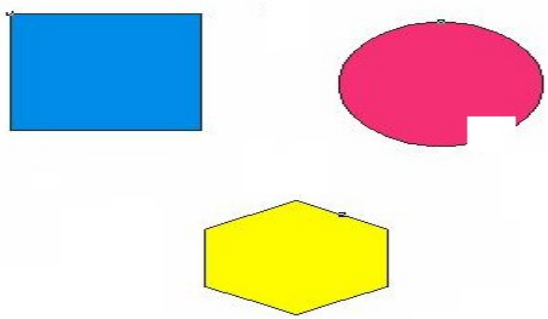


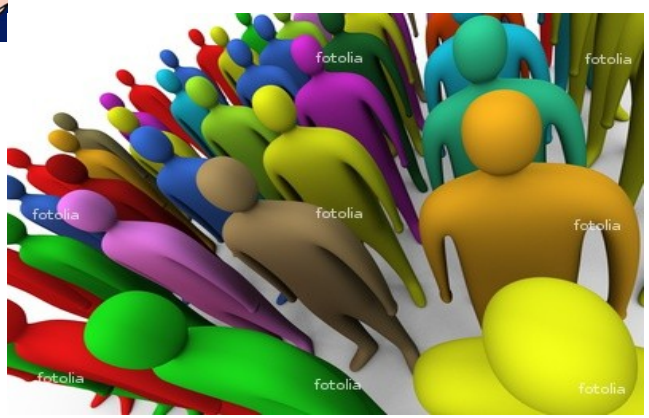
População

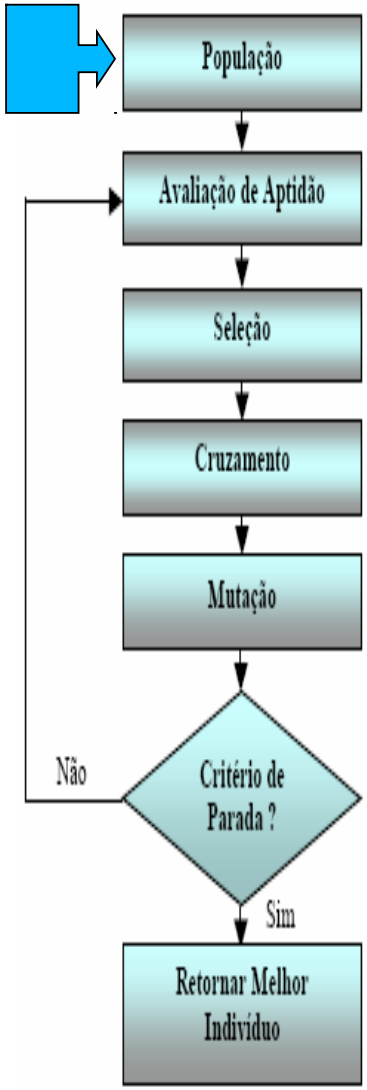
Algoritmos genéticos

A população de um algoritmo genético é o conjunto de indivíduos que estão sendo cogitados como solução

Cada indivíduo é uma possível solução do problema







Indivíduo

Algoritmos genéticos

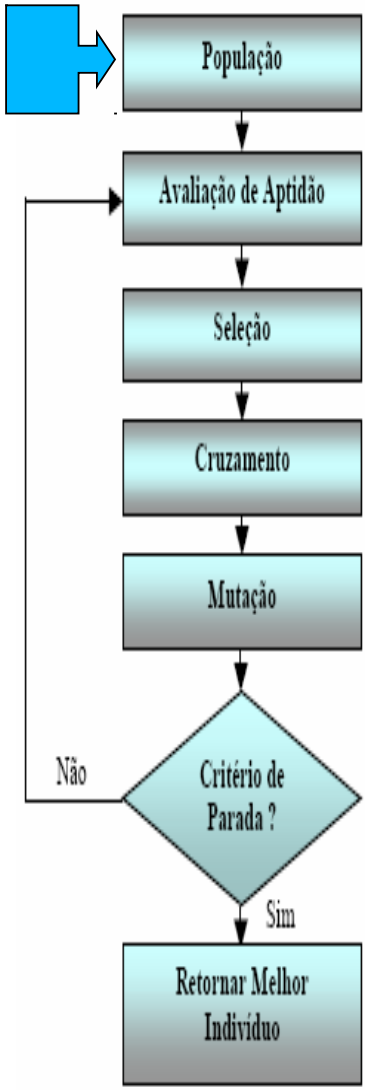
Um indivíduo no AG é um cromossomo

Ou seja, um indivíduo é um **conjunto de atributos da solução**

Geralmente é uma cadeia de bits que representam uma possível solução para o problema

Outras representações são possíveis

Boa representação **depende do problema**



Exemplo: população de tamanho N = 4
geração de indivíduos, com seus cromossomos
Cada elemento do vetor é um gene, um atributo da solução

$$\text{indivíduo } 1 = [\ 1\ 1\ 1\ 0\ 1\ ]$$
$$\text{indivíduo } 2 = [\ 0\ 1\ 1\ 0\ 1\ ]$$
$$\text{indivíduo } 3 = [\ 0\ 0\ 1\ 1\ 0\ ]$$
$$\text{indivíduo } 4 = [\ 1\ 0\ 0\ 1\ 1\ ]$$

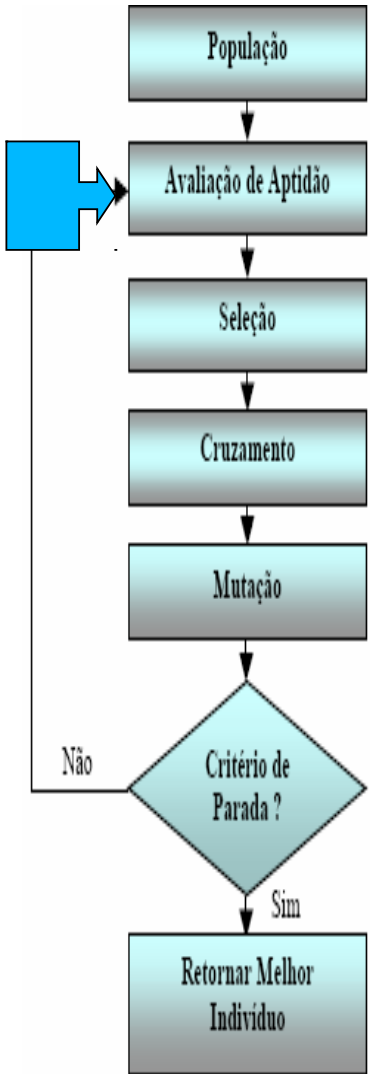
Função de aptidão

Algoritmos genéticos

A função de avaliação, função de *fitness*, que determina uma nota a cada indivíduo
Esta nota avalia quão boa é a solução que este indivíduo representa

Por exemplo: o objetivo de um AG pode ser maximizar o número de 1s

Indivíduos	Função de aptidão (fitness)
1 1 1 0 1	4
0 1 1 0 1	3
0 0 1 1 0	2
1 0 0 1 1	3



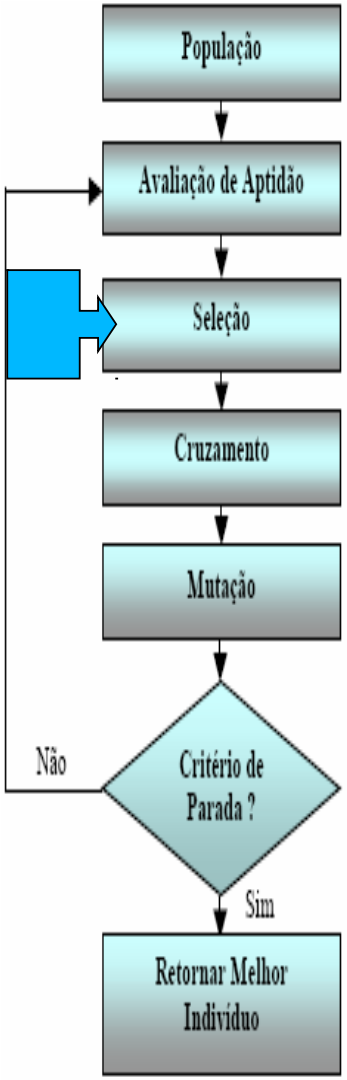
Seleção

Algoritmos genéticos

De acordo com a teoria de Darwin, o melhor sobrevivente para criar a descendência é selecionado
Há muitos métodos pra selecionar o melhor cromossomo
Dentre eles:

- Seleção por roleta
- Seleção por torneio

A seleção dirige o AG para as melhores regiões do espaço de busca



Seleção

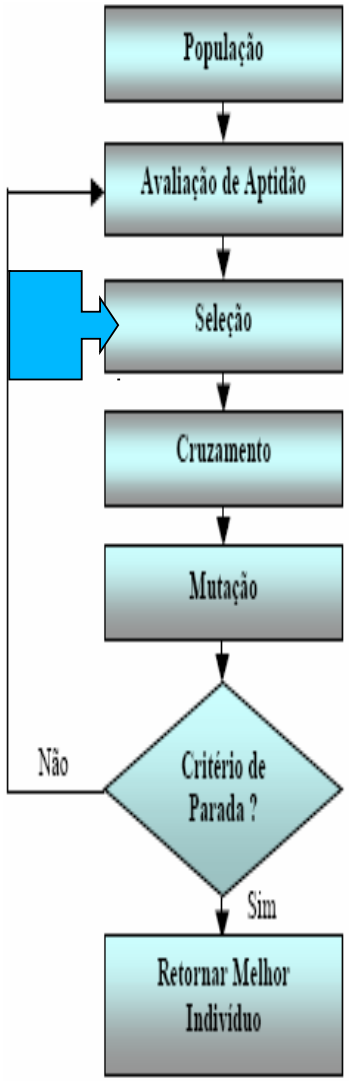
Algoritmos genéticos

No método da roleta cada indivíduo tem ua probabilidade de ser selecionado, proporcional à sua aptidão

cromossomos	x	$f(x^2)$	
$A_1 = 11001$	25	625	54,4%
$A_2 = 01111$	15	225	19,6%
$A_3 = 01110$	14	196	17,1%
$A_4 = 01010$	10	100	8,7%

$$1 * 2^4 + 1 * 2^3 + 0 * 2^2 + 0 * 2^1 + 1 * 2^0 = 16 + 8 + 0 + 0 + 1 = 25$$

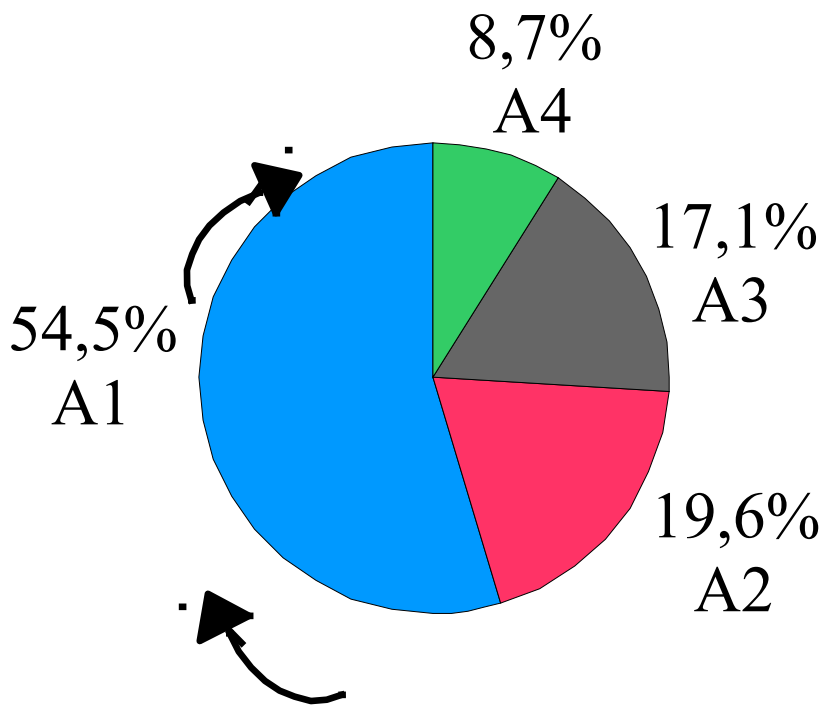
Probabilidade de seleção: $p_i = \frac{f(x_i)}{\sum_{j=1}^{N} f(x_j)}$

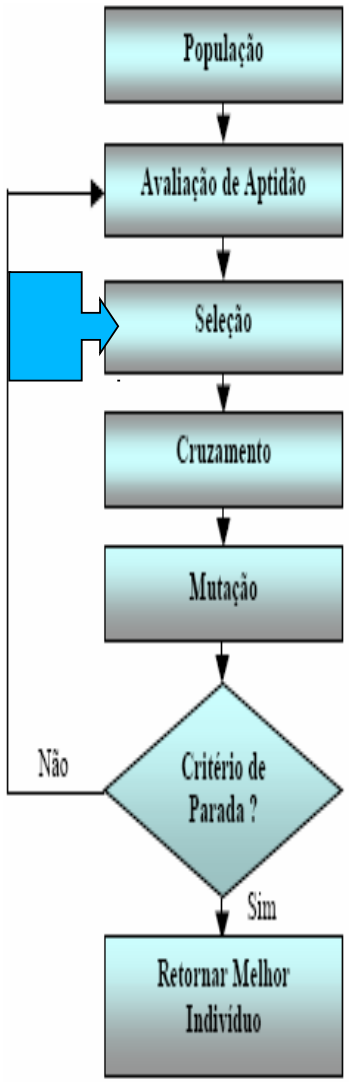


Seleção por roleta

Algoritmos genéticos

Para visualizar este método considere um círculo dividido em N regiões (tamanho da população), onde a área de cada região é proporcional à aptidão do indivíduo

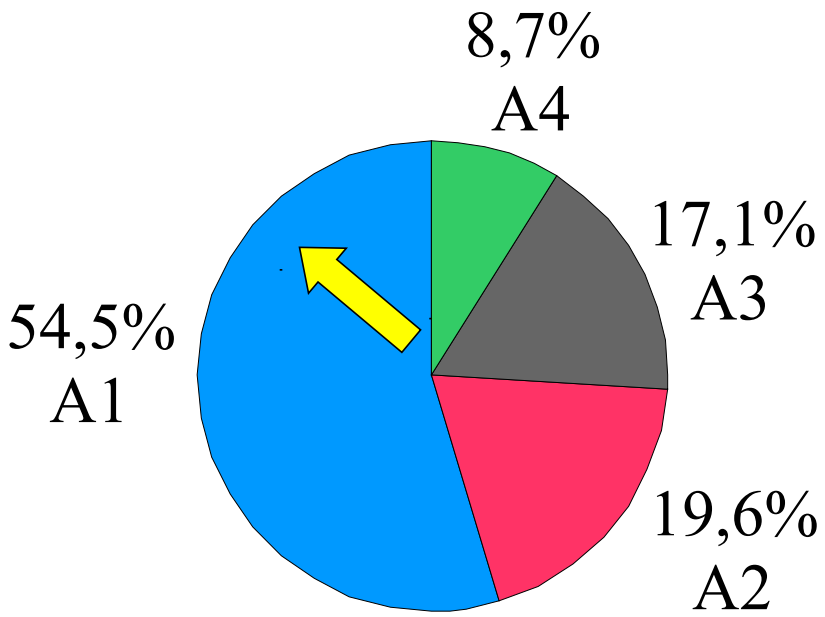


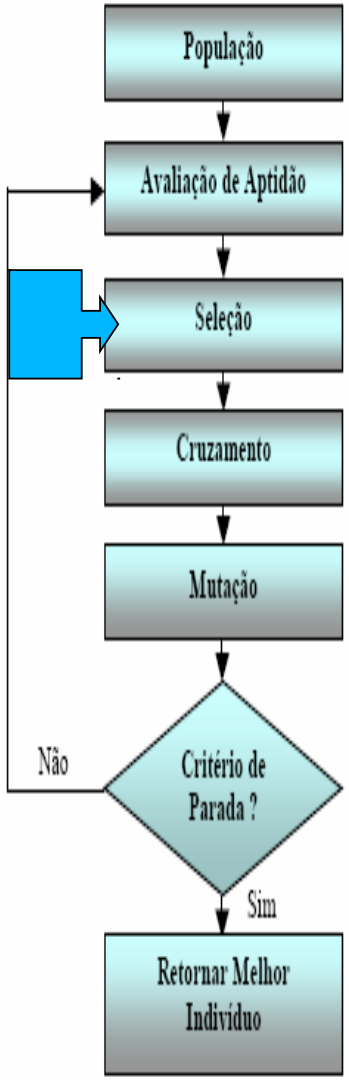


Seleção por roleta

Algoritmos genéticos

Coloca-se sobre este círculo uma "roleta"
A roleta é girada um determinado número de vezes dependendo do tamanho da população
São escolhidos como indivíduos que participarão da próxima geração, aqueles sorteados na roleta





Operadores genéticos

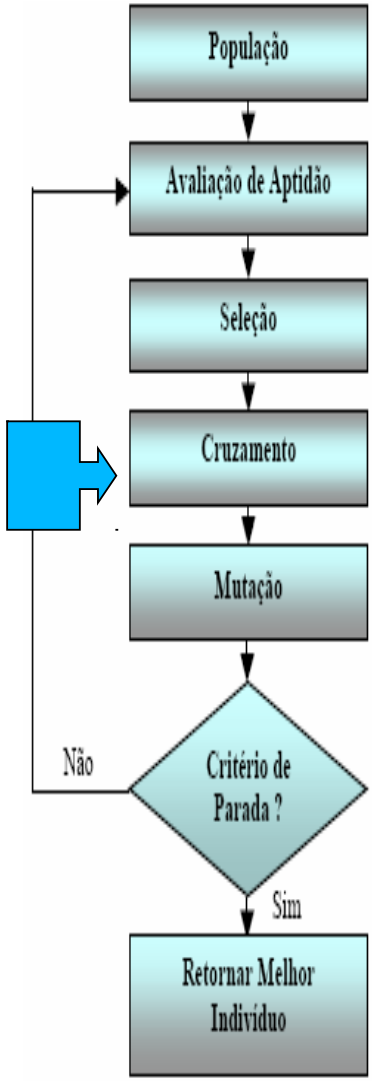
Algortimos genéticos

Um conjunto de operações é necessário para que, dada uma população, se consiga **gerar populações** sucessivas que (espera-se) **melhorem sua aptidão com o tempo**

Estas operações são os operadores genéticos:

- Cruzamento
- Mutação

Os operadores genéticos permitem **explorar** áreas desconhecidas do espaço de busca



Cruzamento

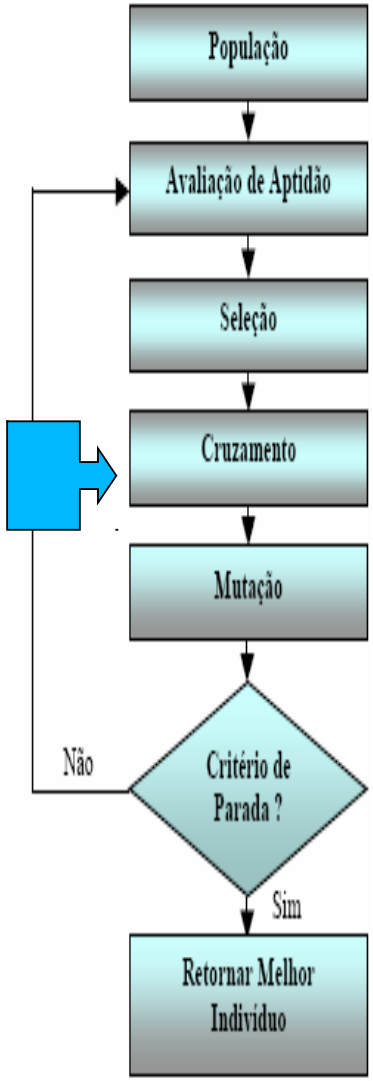
Algoritmos genéticos

O operador *crossover*(cruzamento) cria novos indivíduos, misturando características de dois indivíduos pais

O resultado desta operação é um indivíduo que potencialmente **combine as melhores características** dos indivíduos usados como base

Alguns tipos de cruzamento são:

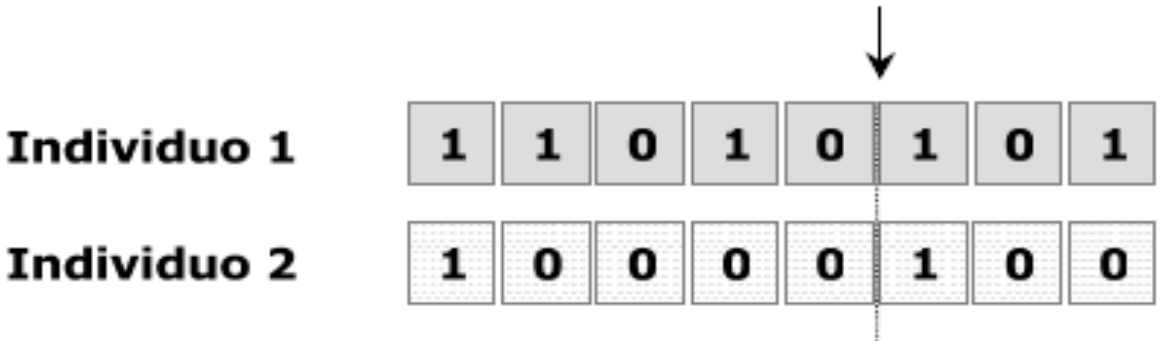
- Cruzamento de um ponto
- Cruzamento em dois pontos

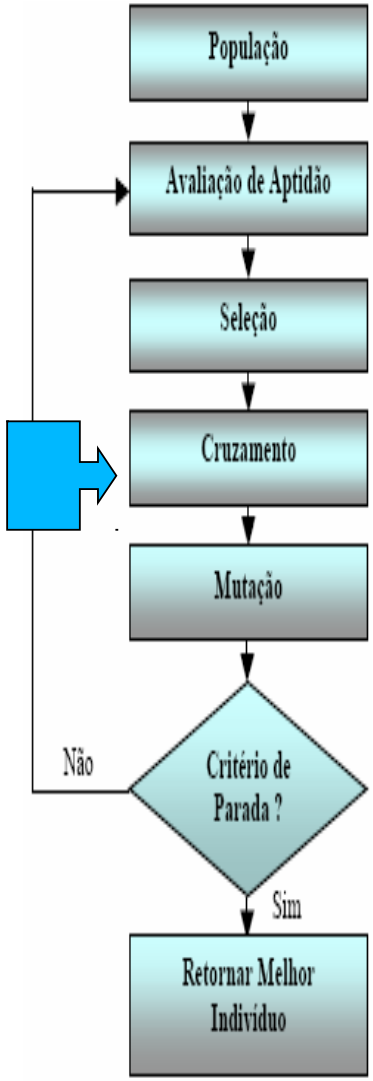


Cruzamento de um ponto

Algoritmos genéticos

No cruzamento de um ponto divide-se cada progenitor em duas partes, em uma localidade k (escolhida aleatoriamente)

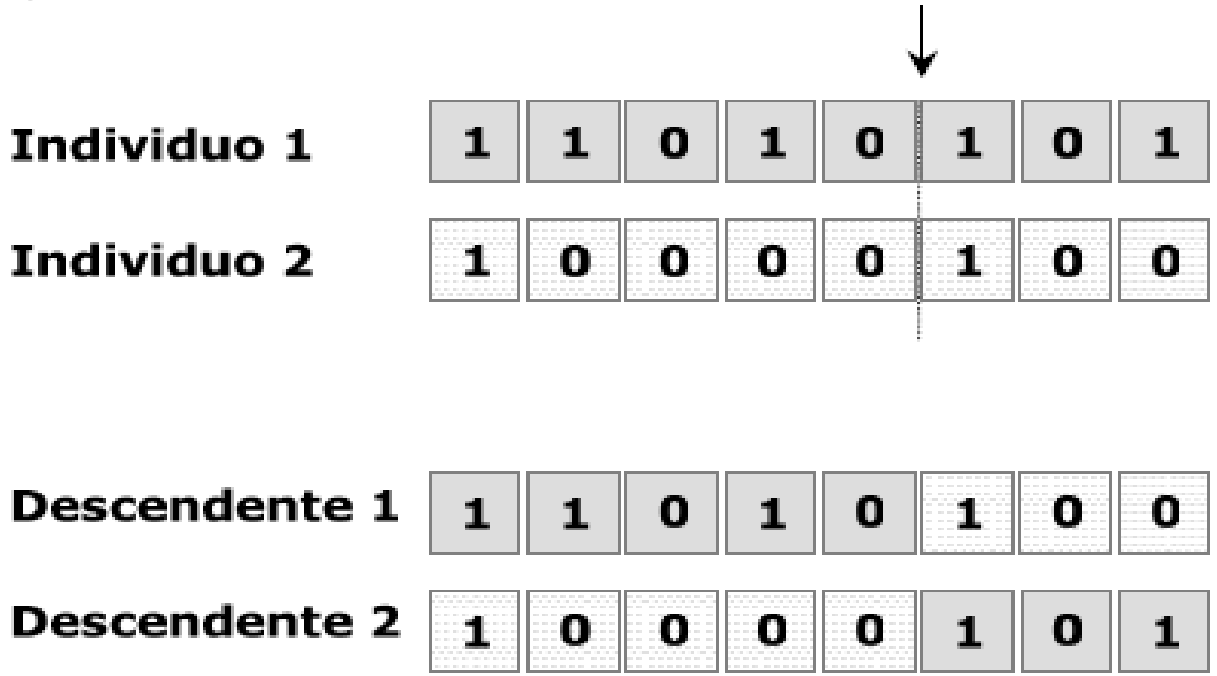


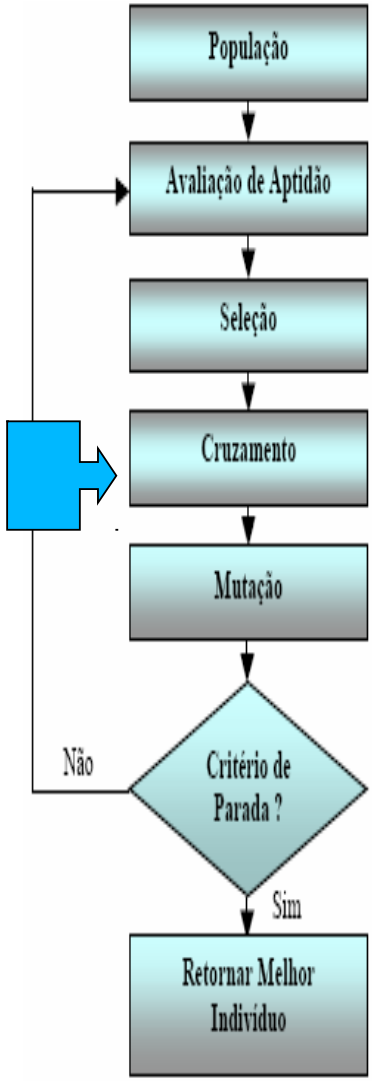


Cruzamento de um ponto

Algoritmos genéticos

O descendente 1 consiste em genes 1 a $k-1$ do progenitor 1, e genes k a n do progenitor 2
O descendente 2 é o "reverso"



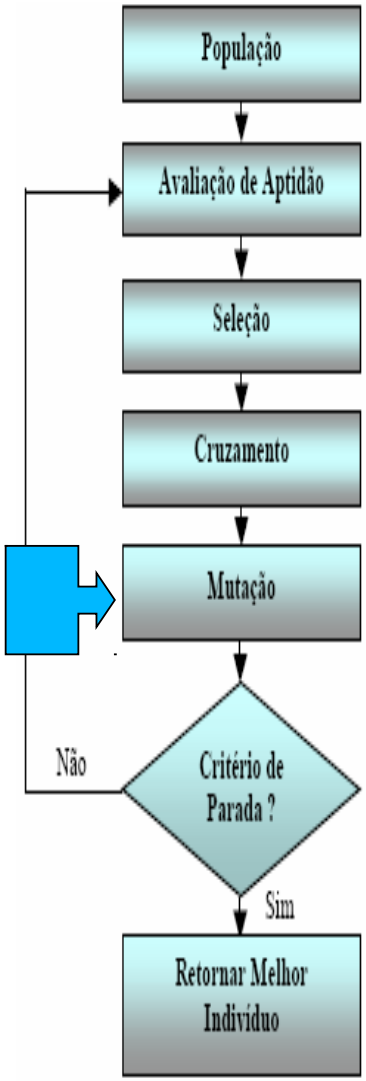


Mutação

Algoritmos genéticos

A mutação modifica aleatóriamente alguma característica do indivíduo, sobre o qual é aplicada
O operador de mutação é necessário para a **introdução e manutenção da diversidade** genética da população
Desta forma, a mutação assegura que a probabilidade de se **chegar a qualquer ponto do espaço de busca**, possivelmente, não será zero

Individuo	1	1	0	1	0	1	0	1
Individuo Mutado	1	1	0	0	0	1	0	1



Gerações

Algoritmos genéticos

Algoritmo é iterado até algum **critério de parada**
A cada passo, um novo conjunto de indivíduos é gerado a partir da população anterior
A este novo conjunto da-se o nome de geração
Com a criação de uma grande quantidade de gerações que é possível obter resultados dos AGs

Algoritmo

Algoritmos genéticos

```
1   Algoritmo_genetico
2     p = tamanho da populacao
3     r = taxa de cruzamento
4     m = taxa de mutacao
5     P = lista de individuos
6     PS = lista de individuos selecionados
7
8     P = gerar aleatoriamente p individuos
9     Para cada i em P, computar Aptidao(i)
10    Enquanto criterio_parada nao eh atingido
11      Selecionar p membros de P pra reproducao
12      Aplicar cruzamento a pares de individuos selecionados
            segundo taxa r, adicionando filhos em PS
13      Realizar mutacao em membros PS, segundo taxa m
14      P = PS
15      Para cada i em P, computar Aptidao(i)
16    Retornar o individuo de P com maior aptidao
```

Exemplo

Algoritmos genéticos

Codificando o problema Ex.: problema 8 rainhas

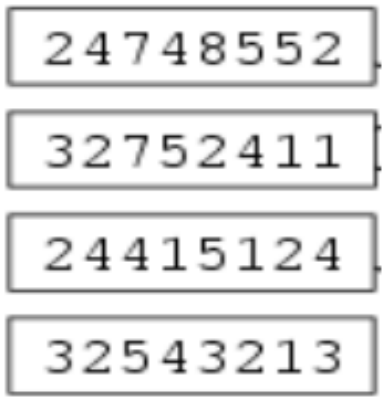
- Cada solução candidata deve especificar posição de 8 rainhas, em coluna com 8 quadrados
  - $8 * log_2(8) = 24$ bits de codificação binária

Exemplo

Algoritmos genéticos

(a) Gerando população inicial 8 dígitos, com valores de 1 a 8

- Exemplo: população com 4 cadeias de 8 dígitos
  - representam soluções candidatas de 8 rainhas



Exemplo

Algoritmos genéticos

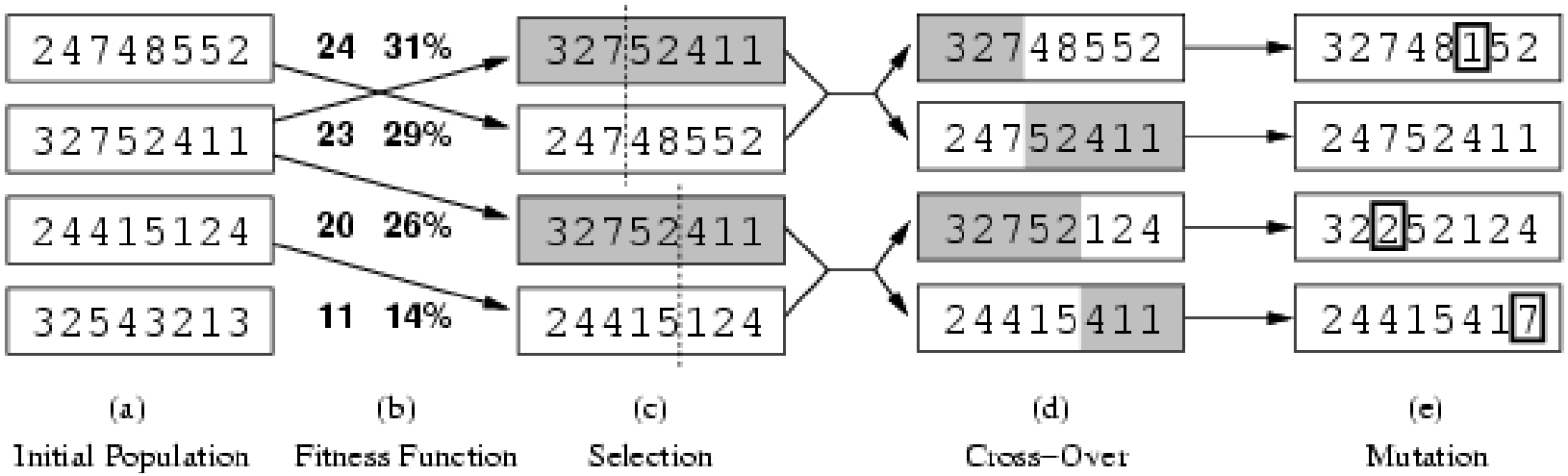
(b) Avaliação

Função de avaliação (aptidão)

- Deve retornar valores maiores para soluções melhores
- Ex.: 8 rainhas: número de pares de rainhas não-atacantes
  - Valor 28 para uma solução
    - (min $= 0$, max $= 8 * \frac{7}{2} = 28$)

Exemplo

Algoritmos genéticos



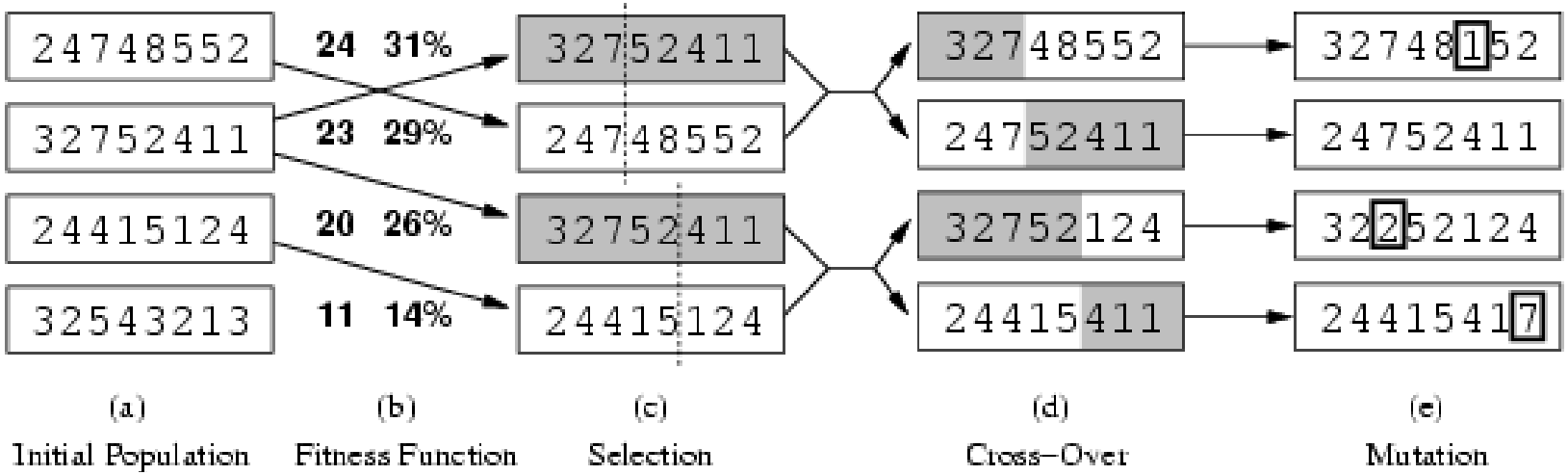
(c) Seleção

Proporcional à aptidão do indivíduo

- Vários métodos
- Todos tendem a privilegiar indivíduos mais aptos
  - $\frac{24}{(24+23+20+11)} = 31\%$ no exemplo
  - $\frac{23}{(24+23+20+11)} = 29\%$ etc

Exemplo

Algoritmos genéticos



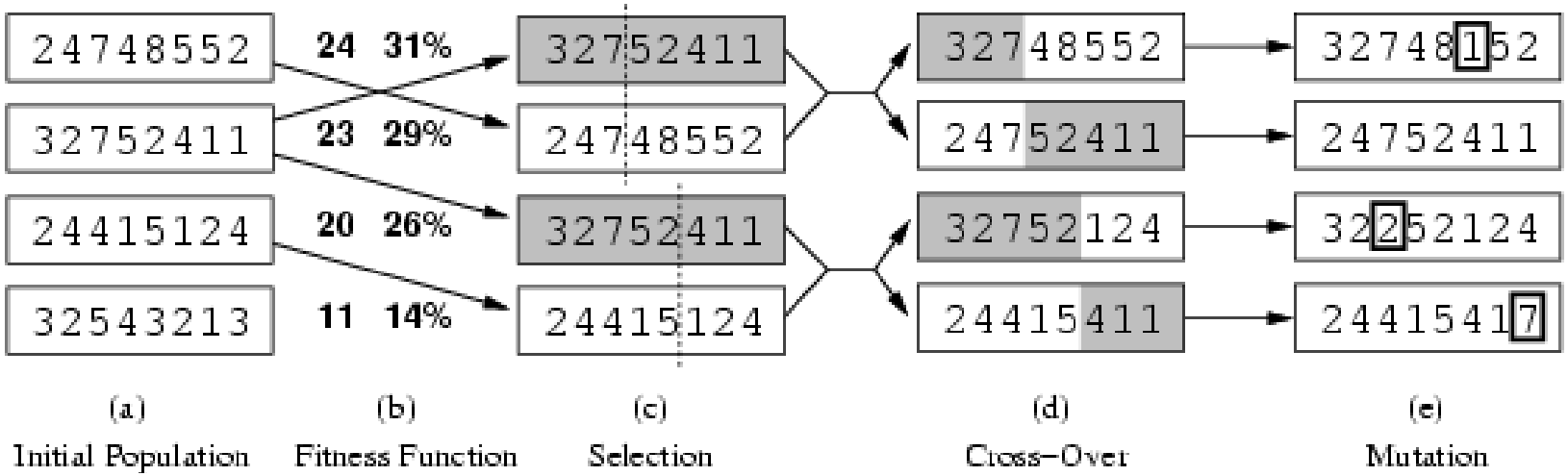
(d) Cruzamento

indivíduos selecionados formam pares

Operador de cruzamento combina pares

- ponto de cruzamento gerado ao acaso

Algoritmos Genéticos



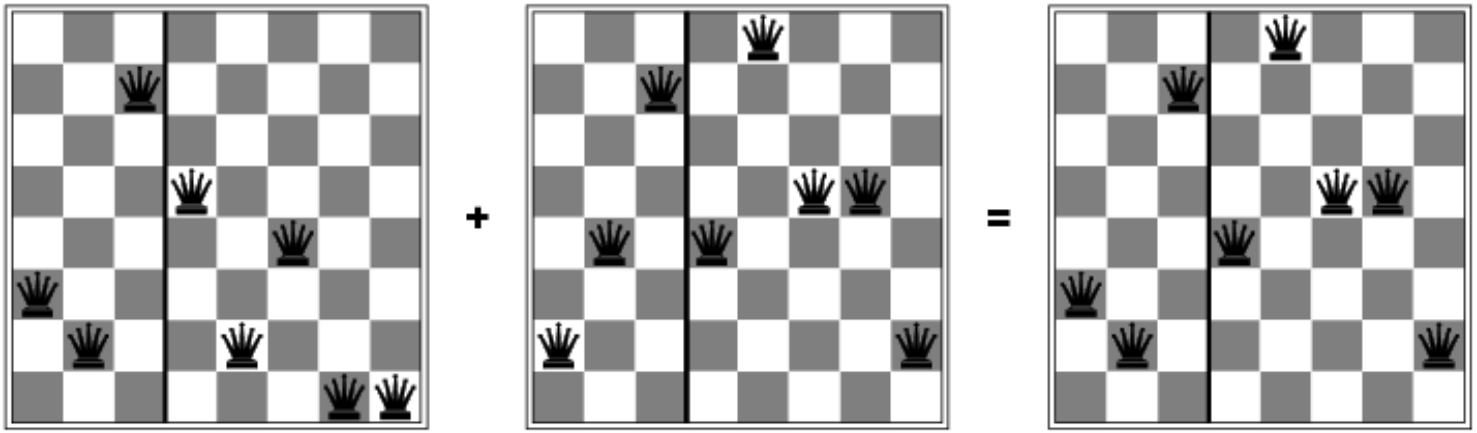
(e) Mutação

Mudança aleatória do valor de um gene

- Com pequena probabilidade introduz variações aleatórias
  - Permitindo soluções pularem para **diferentes partes do espaço de busca**

Algoritmos Genéticos

Cruzamento



Observações

Se o AG estiver corretamente implementado, a população evolui em gerações sucessivas

Aptidão do melhor indivíduo e do indivíduo médio aumentam em direção a um ótimo global

Algoritmo genético

Exercício

Considere o problema das 4-rainhas

- Considere a seguinte popoulação inicial do AG

  1 1 2 2
  1 2 3 4
  4 3 2 1
  2 3 1 4

- Decodifique esses indivíduos e aplique um passo do algoritmo genético a eles, usando:
  - Cruzamento de um ponto entre os pares de indivíduos 1-2 e 3-4
  - Mutação somente no último gene do primeiro indivíduo produzido pelo cruzamento

Buscas apresentadas

Hill Climbing e Algoritmos Genéticos

Conseguem lidar com espaços de busca que sejam muito grandes (e.g., $10^{(1000)}$)

Geralmente são os melhores algoritmos quando se tem pouca ou nenhuma informação global

- Se conhece pouco o problema

Animações Busca

Subida de Encosta

http://www.kramer.me.uk/robin/NetLogo/hill-climbing%20algorithm.html

Algoritmo Genético

http://math.hws.edu/xJava/GA/

http://www.obitko.com/tutorials/genetic-algorithms/example-function-minimum.php

http://www.obitko.com/tutorials/genetic-algorithms/example-3d-function.php

http://www.obitko.com/tutorials/genetic-algorithms/tsp-example.php

Referências

- Livros
  - Russel e Norvig: Inteligência Artificial, cap 3
  - R. Linden: Algoritmos Genéticos
  - Z. Michelewicz: Genetic Algorithm + Data Structures = Evolution Programs
- Slide
  - Ana Carolina Lorena. Unifesp
  - Richard Khoury, University of Waterloo
  - UFPE
  - Cornell University
  - Maria das Graças B. Marieto. UFABC