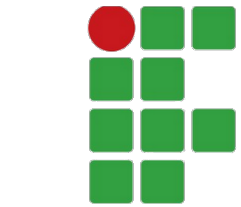
Inteligência Artificial

Metaheurísticas

Felipe Augusto Lima Reis

felipe.reis@ifmg.edu.br



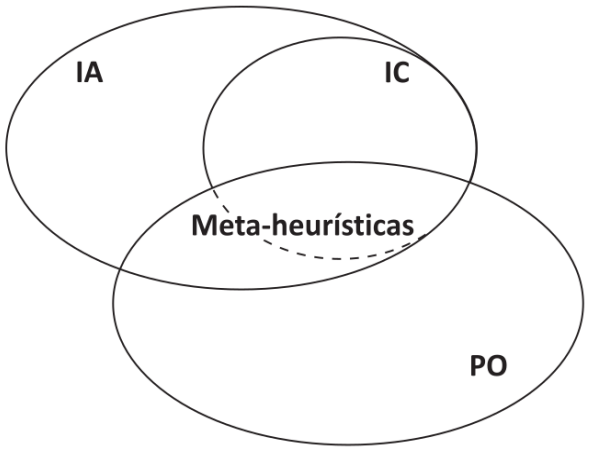
INSTITUTO FEDERAL
Minas Gerais

Sumário

Introdução

Contexto

- As heurísticas e metaheurísticas são um campo comum da Pesquisa Operacional, da IA e IC;
- Surgiram como alternativa aos métodos exatos para resolução de problemas de otimização de alta complexidade
  - Problemas que não podem ser solucionados em tempo polinomial (NP e NP-completos) [Belfiore and Fávero, 2013]



Fonte: Silva Neto e Becceneri (2009) apud [Belfiore and Fávero, 2013]

Conceitos

Conceitos - Otimização

- Otimização refere-se à escolha do melhor elemento em um conjunto de alternativas [Luzia and Rodrigues, 2009];
- Um problema de otimização consiste em maximizar ou minimizar uma função linear de variáveis de decisão, sujeita a um conjunto de restrições [Belfiore and Fávero, 2013]
- O resultado do problema de otimização é a produção de uma solução ótima
  - Corresponde ao melhor valor possível para um dado problema, segundo critérios estabelecidos previamente (restrições).

Conceitos - Pesquisa Operacional

INSTITUTO FEDERAL
Minas Gerais

- Segundo [Belfiore and Fávero, 2013], técnicas para solução de problemas de Pesquisa Operacional podem ser divididas em 3 grandes grupos:
  - Modelos Determinísticos
    - Programação Linear, Não Linear, Dinâmica e Em Redes;
  - Modelos Estocásticos
    - Teoria das Filas e dos Jogos, Prog. Dinâmica Estocástica;
  - Outras técnicas (heurísticas e metaheurísticas)
    - Inteligência Artificial (IA), Inteligência Computacional;
- Segundo [Luke, 2013], heurísticas e metaheurísticas correspondem a um tipo de otimização estocástica.

Conceitos - Heurísticas

INSTITUTO FEDERAL
Minas Gerais

"Heurística pode ser definida como um procedimento de busca guiada pela intuição, por regras e ideias, visando encontrar uma boa solução." [Belfiore and Fávero, 2013]

Conceitos - Metaheurísticas

INSTITUTO FEDERAL
Minas Gerais

"Metaheurística é a combinação de procedimentos de busca com estratégias de mais alto nível, incluindo intensificação e diversificação, buscando escapar de ótimos locais e encontrar soluções muito próximas do ótimo global, porém sem garantia da otimalidade." [Belfiore and Fávero, 2013]

Metaheurísticas

Metaheurísticas

INSTITUTO FEDERAL
Minas Gerais

- Segundo [Luke, 2013] e [Luzia and Rodrigues, 2009], metaheurísticas são aplicadas em problemas de otimização sobre os quais há poucas informações:
  - Não se sabe previamente qual a solução ótima esperada;
  - Existem poucas ou nenhumas heurísticas disponíveis;
  - Força-bruta é inadequada, devido ao espaço de solução ser muito grande;
  - A solução candidata pode ser testada quanto a sua qualidade.

Metaheurísticas Construtivas

- Soluções construtivas são aquelas "construídas de forma iterativa através da adição de componentes a uma solução inicial nula, sem *backtracking*, até que uma solução completa seja encontrada [Luzia and Rodrigues, 2009]";
- Uma metaheurística construtiva estabelece estratégias para a construção de uma solução, de modo que em cada passo, ela adiciona um elemento à solução parcial, de forma a obter os melhores resultados [Sucupira, 2004].

Metaheurísticas baseadas em população

INSTITUTO FEDERAL
Minas Gerais

- Métodos baseados em população contém um conjunto de soluções candidatas ao invés de uma solução única;
- Ao contrário de outros métodos, quando implementados de forma paralela, os algoritmos populacionais possibilitam que soluções candidatas afetem umas às outras.
- Soluções boas são propagadas e soluções ruins são rejeitadas, fazendo com que o algoritmo tenha convergência em direção a melhores soluções [Luke, 2013].

Metaheurísticas baseadas em população

INSTITUTO FEDERAL
Minas Gerais

- Alguns métodos populacionais tem inspiração na biologia
  - Por utilizar conceitos de biologia, genética e evolução, essas técnicas são conhecidos como Computação Evolucionária (EC);
  - Algoritmos relacionados a essas técnicas são conhecidos como Algoritmos Evolucionários (EA);
  - Segundo [Luke, 2013], algoritmos evolucionários podem ser classificados em dois tipos principais[1]:
    - Algoritmos geracionais: alteram uma população inteira a cada iteração;
    - Algoritmos de estado estacionário: atualizam uma amostra de candidatos da solução a cada época.

[1] Existem outras classificações e sub-classificações de algoritmos populacionais.

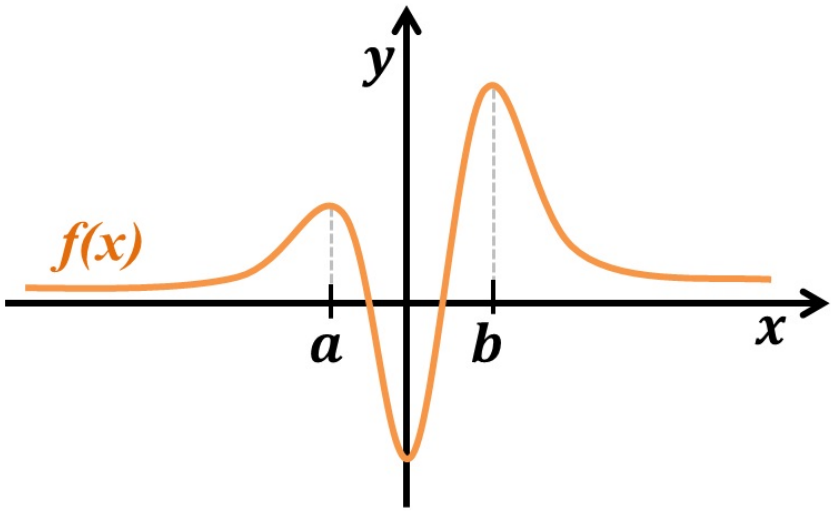
Busca Local (Otimização)

INSTITUTO FEDERAL
Minas Gerais

- Método heurístico para solução computacional de problemas de otimização difícil;
- Pode ser usado para encontrar uma solução que maximize um critério entre várias soluções candidatas;
- A solução inicia-se com uma configuração inicial (aleatória) e é modificada de forma a mover-se pelo espaço de soluções
  - Pequenas modificações são feitas nas soluções potenciais até atingir um critério de parada [Coppin, 2004].

Pontos de Máximo e Mínimo

- Pontos de máximos e mínimos são os pontos de picos e de depressões da função;
- Podemos ter máximos/mínimos locais e absolutos (globais);
- Esses pontos, em funções otimização são conhecidos como ótimos locais e ótimos globais.

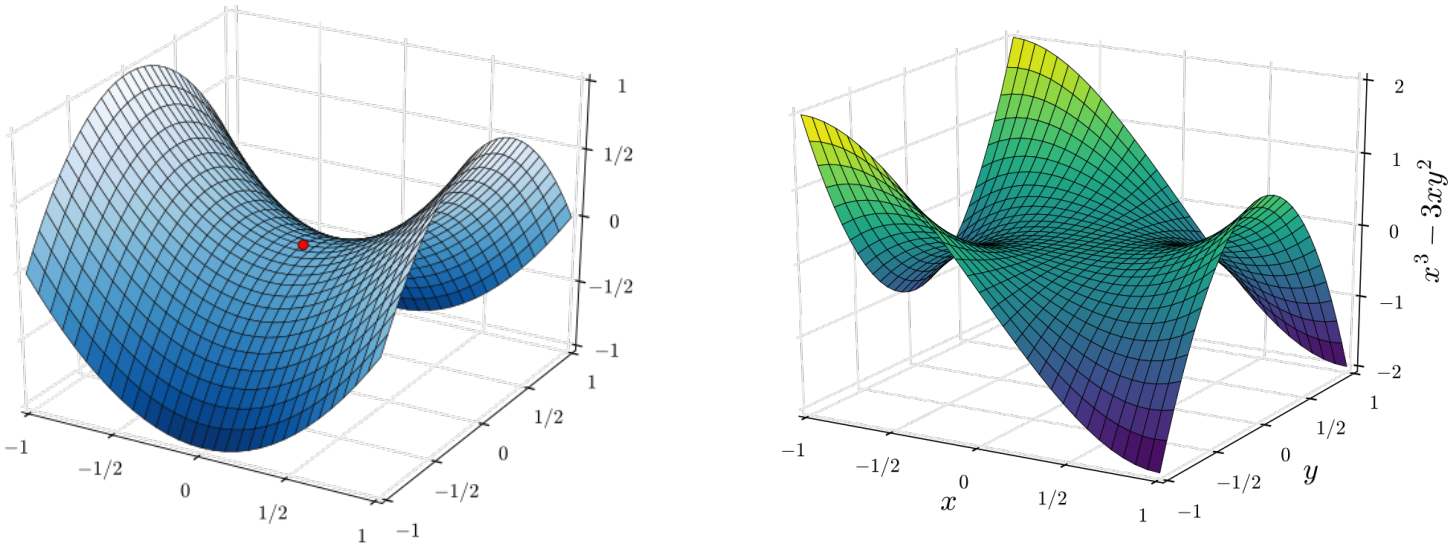


Fonte: [Dicas de Cálculo, 2020]

Pontos de Sela

INSTITUTO FEDERAL
Minas Gerais

- Um ponto de sela (*saddle point*) é o ponto sobre uma superfície no qual a declividade é nula;
- É o ponto sobre na qual a elevação é máxima em uma direção e mínima em outra;
- Não corresponde ao ponto de mínimo tampouco de máximo.



Fonte: [Wikipedia contributors, 2020]

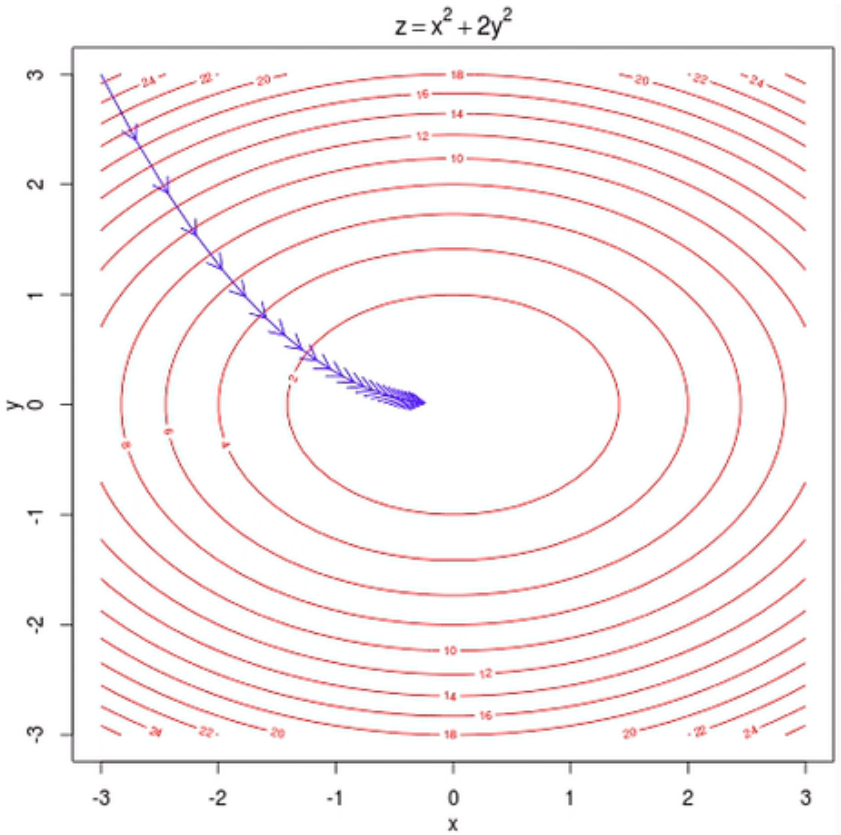
Método do Gradiente

INSTITUTO FEDERAL
Minas Gerais

- O método do gradiente é um método numérico usado para maximização ou minimização de uma função objetivo [Luke, 2013]
  - Método do máximo declive (*gradient descent*) busca minimizar o erro esperado;
  - Método do *gradient ascent* busca encontrar o valor máximo de uma função;
- O método busca seguir o caminho mais íngreme na superfície que representa a função objetivo [Coppin, 2004] [Luke, 2013].

Método do Gradiente

- No exemplo abaixo, o método segue mais íngreme na superfície da função objetivo.



Fonte: [Hoang Duong - Hoang Duong blog, 2015]

Hill-Climbing

Hill-Climbing[2]

- Hill Climbing é um tipo de busca local informada, relacionada ao método do gradiente [Coppin, 2004] [Luke, 2013];
- Tipo de método de otimização local, uma vez que tenta otimizar um conjunto de valores e frequentemente encontra o máximo local, ao invés do máximo global [Coppin, 2004];
- Não armazena o caminho percorrido até a solução corrente e sim a solução propriamente dita como estado [Luzia and Rodrigues, 2009].

[2] Tradução literal: escalada de montanha

Hill-Climbing

- Apesar de relacionado ao método do gradiente, o Hill-Climbing não necessita de conhecer o gradiente ou a direção;
- As soluções candidatas são testadas na região do candidato atual;
- Caso uma solução candidata seja melhor que a solução atual, ela é adotada;
- O algoritmo "escala a montanha" até atingir o ótimo local [Luzia and Rodrigues, 2009] [Luke, 2013].

Hill-Climbing

- O algoritmo inicia com uma solução randômica, potencialmente ruim;
- Iterativamente, efetua pequenas modificações na solução atual, em busca de melhorar o resultado da função objetivo;
- O algoritmo termina quando não encontra nenhuma melhoria possível em uma iteração;
- Idealmente a solução obtida é ótima[3], porém não há garantia de otimalidade [Luzia and Rodrigues, 2009].

[3] O Hill Climbing produzirá, na maioria dos casos, ótimos locais.

Hill Climbing

INSTITUTO FEDERAL
Minas Gerais

- O pseudo-algoritmo do Hill Climbing pode ser visto abaixo;
- É possível observar que o algoritmo é extremamente simples e pode ser implementado rapidamente;
- O método *"NovaSolução"* (linha 3), pode ser implementado como um simples transformação estocástica (aleatória) a partir do valor anterior [Luzia and Rodrigues, 2009].

```
1: S ← solução inicial
2: repita
3:     R ← NovaSolução(S)
4:     se (Qualidade(R) > Qualidade(S)) então
5:         S ←R
6: até que S seja a solução ideal ou o tempo tenha se esgotado
7: devolva S
```

Hill Climbing, versão básica

Fonte: [Luzia and Rodrigues, 2009]

Steepest Ascent Hill Climbing

INSTITUTO FEDERAL
Minas Gerais

- Nesta melhoria são avaliados os sucessores da solução atual, para escolha daquela que oferecer o melhor resultado [Luzia and Rodrigues, 2009] [Luke, 2013];
- O algoritmo pode tomar *n* direções aleatórias e escolher aquela cujo crescimento (ou decrescimento) é maior.

```
01: n ← número de extensões a serem geradas
02: S ← solução candidata inicial qualquer
03: repita
04:     R ← NovaSolução( S )
05:     repita n - 1 vezes
06:         W ← NovaSolução( S )
07:         se Qualidade(W) > Qualidade(R) então
08:             R ← W
09:     se Qualidade(R) > Qualidade(S) então
10:         S ← R
11: até que S seja a solução ideal ou o tempo tenha se esgotado
12: devolva S
```

Steepest Ascent Hill Climbing

Fonte: [Luzia and Rodrigues, 2009]

Steepest Ascent Hill Climbing with Replacement

INSTITUTO FEDERAL
Minas Gerais

- Nesta versão, não é avaliado se a solução seguinte é melhor que a atual;
- A solução atual é simplesmente substituída pela seguinte, sem comparação;
- Essa variação tem como objetivo aumentar a área avaliada pelo algoritmo;
- Como ponto negativo, pode não retornar a melhor solução, uma vez que ela pode ser perdida ao longo do tempo [Luke, 2013].

Hill Climbing with Random Restart

INSTITUTO FEDERAL
Minas Gerais

- Esta melhoria busca fazer com que o método cubra uma maior área no espaço de busca;
- O algoritmo executa o Hill Climbing por um certo tempo e então efetua uma busca aleatória, de modo a fugir de máximos locais [Luzia and Rodrigues, 2009] [Luke, 2013].

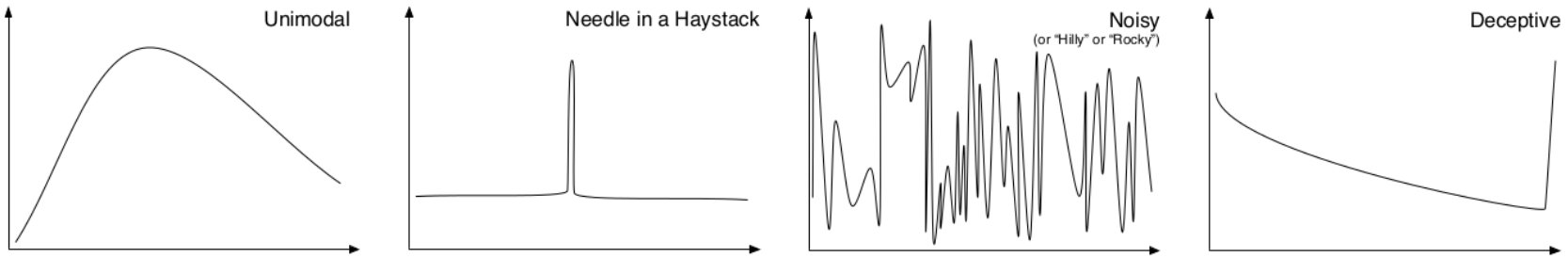
```
01: S ← solução candidata inicial qualquer
02: Melhor ← S
03: repita
04:     período ← NovoTempo()
05:     repita
06:         R ← NovaSolução( S )
07:         se Qualidade(R) > Qualidade(S) então
08:             S ← R
09:     até que S seja ideal, ou que período ou o tempo total tenham se esgotado
10:     se Qualidade(S) > Qualidade(Melhor) então
11:        Melhor ← S
12:     S ← solução candidata aleatória
13: até que S seja a solução ideal ou o tempo tenha se esgotado
14: devolva Melhor
```

Hill Climbing with Random Restart

Fonte: [Luzia and Rodrigues, 2009]

Hill Climbing

- O Hill Climbing e suas variações podem possuir dificuldade de convergência para máximos globais em algumas funções
  - Dependendo da aplicação, uma análise do cenário e da função a ser solucionada pode auxiliar na escolha do algoritmo;
  - Melhorias do Hill Climbing, em geral, tem desempenho superior ao algoritmo original.



Fonte: [Luke, 2013]

Vantagens e Desvantagens

INSTITUTO FEDERAL
Minas Gerais

- Vantagens:
  - Fácil de ser implementado;
  - Pode ser utilizado como base para construção de métodos mais sofisticados [Luzia and Rodrigues, 2009];
- Desvantagens:
  - Possibilidade de ficar preso em máximos locais;
  - Baixa efetividade em "regiões planas";
  - Efetividade do método é dependente da função em que está sendo aplicado [Luzia and Rodrigues, 2009].

Recozimento Simulado (Simulated Annealing)

Recozimento Simulado (*Simulated Annealing*)

- Fundamentado em uma analogia com a termodinâmica;
- Recozimento (*annealing*), em metalurgia, corresponde ao processo de "aquecer um metal até o ponto de fusão e então resfriá-lo lentamente, permitindo que as moléculas alcancem uma configuração de baixa energia e formem uma estrutura cristalina, livre de defeitos" [Luzia and Rodrigues, 2009].

Recozimento Simulado (*Simulated Annealing*)

INSTITUTO FEDERAL
Minas Gerais

- Foi desenvolvido em 1983 por Kirkpatrick et al. [Souza, 2011] [Kirkpatrick et al., 1983];
- O algoritmo é considerado uma estratégia de busca local [Luzia and Rodrigues, 2009];
- A cada iteração, são gerados valores para duas soluções, a atual e uma escolhida, que são comparadas
  - Soluções melhores que a atual são sempre aceitas;
  - Uma fração de soluções piores que a atual são aceitas, permitindo que o algoritmo escape de máximos locais [Luzia and Rodrigues, 2009] [Luke, 2013].

Simulated Annealing

- Para um problema de minimização, a variação de valor da função objetivo ao mover-se de uma solução S para uma solução vizinha candidata S' é definida como:

$$\Delta S = f(S') - f(S)$$

- Temos dois valores possíveis para ΔS
  - $\Delta < 0$: método aceita o movimento e a solução vizinha passa a ser a nova solução corrente
  - $\Delta \geq 0$: a solução vizinha candidata poderá ser aceita com uma probabilidade $e^{-\Delta/t}$, onde t é uma parâmetro denominado temperatura [Souza, 2011].

Simulated Annealing

INSTITUTO FEDERAL
Minas Gerais

- A probabilidade de aceitar uma nova solução pode ser dada por: [Luke, 2013] e [Luzia and Rodrigues, 2009]

$$P(t, R, S) = e^{\left(\frac{Qualidade(R) - Qualidade(S)}{t}\right)}$$

onde

- t: parâmetro do algoritmo, correspondente à temperatura de recozimento, definida a cada iteração;
- R: solução de qualidade inferior;
- S: solução atual, de qualidade superior;
- e: constante; número de Euler;

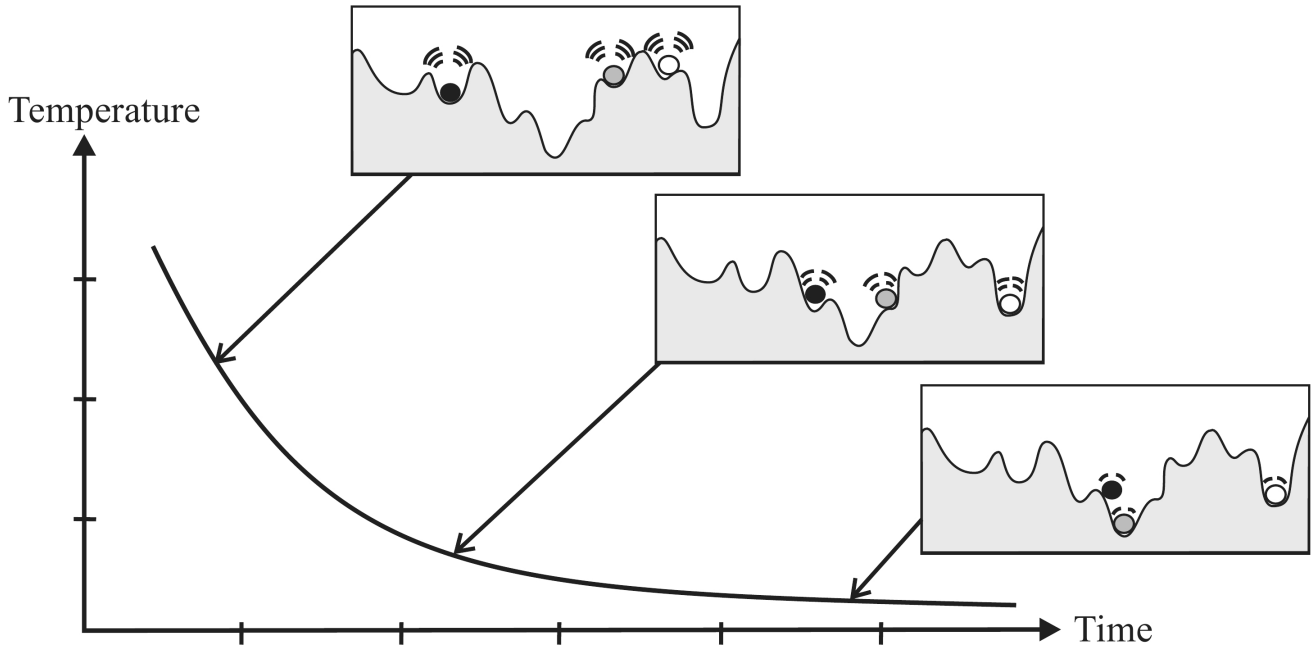
Simulated Annealing

- A taxa de decrescimento ("temperatura" t) é chamada de *schedule* do algoritmo e pode ser ajustado (*fine-tuning*);
- O parâmetro t inicia com um valor alto t_0, que vai gradualmente se reduzindo (para $t \geq 0$);
- A cada iteração, a temperatura é reduzida, o que diminui a probabilidade de escolha de uma solução menos promissora e aumenta a tendência de se melhorar a solução atual;
- Quando a temperatura atinge o valor zero[4], o sistema é "congelado" e o mínimo de energia do sistema é atingido [Coppin, 2004] [Luke, 2013] [Luzia and Rodrigues, 2009].

[4] Na prática, em alguns sistemas, a temperatura não precisa atingir o valor zero. Quando a temperatura fica baixa o suficiente para não permitir a troca da solução, a execução do algoritmo pode ser finalizada [Souza, 2011].

Simulated Annealing

- O possível comportamento e a influência da temperatura pode ser visto na imagem abaixo;
- Como o decrescimento nem sempre é "perfeito", o aceite de soluções piores que a atual podem auxiliar na convergência.



Fonte: [da Silva, 2017]

Vantagens e Desvantagens

INSTITUTO FEDERAL
Minas Gerais

- Vantagens:
  - Se a temperatura t é reduzida de forma suficientemente lenta, então o sistema pode atingir o equilíbrio;
  - O algoritmo tem garantia de convergência (desde que a temperatura seja reduzida de forma suficientemente lenta) [Luzia and Rodrigues, 2009];
- Desvantagens:
  - Apesar da convergência ser garantida, ela pode ser muito lenta;
  - O "resfriamento rápido" pode não garantir a convergência da solução [Luzia and Rodrigues, 2009].

Algoritmos Genéticos

Introdução

- Algoritmos Genéticos (AG) correspondem a uma conjunto de métodos entre a coleção de algoritmos conhecidos como Algoritmos Evolucionários [Luke, 2013];
- AGs foram desenvolvidos em 1975, por John Holland [Luzia and Rodrigues, 2009] [Bozorg-Haddad et al., 2017]
  - AGs, posteriormente, passaram a ser relacionados com os termos de computação evolucionária;
  - Diversas variações do AG original foram publicadas na literatura, com propostas de evoluções e melhorias.

Introdução

- Os algoritmos genéticos são inspirados na Teoria da Evolução, proposta por Charles Darwin
  - A teoria indica que o indivíduos de uma geração são selecionados a partir dos indivíduos mais aptos entre os organismos ameaçados por predadores e/ou riscos ambientais;
  - Seus descendentes herdam suas características;
  - Mutações e combinações entre indivíduos podem aumentar as chances de persistência da espécie no longo prazo [Bozorg-Haddad et al., 2017].

Conceitos

- Para solução de um problema, um AG gera uma quantidade de n indivíduos
  - Esses indivíduos são denominados cromossomos;
  - Um cromossomo é composto de genes;
  - Cada gene pode ser interpretado como uma variável de decisão;
- Cada indivíduo gerado pelo AG corresponde a uma possível solução de um problema de otimização [Coppin, 2004] [Bozorg-Haddad et al., 2017].

A proposta original de AGs proposta por Holland utilizava uma string de bits, conhecida como cromossomos, compostos por genes [Coppin, 2004].

Conceitos

- A população pode ser definida como um conjunto de indivíduos (cromossomos);
- *Fitness* (aptidão) corresponde ao quanto um indivíduo é capaz de interagir com o objetivo do problema
  - A medida de *fitness* é uma extensão das característica físicas (fenótipo) e genéticas (genótipo) [Coppin, 2004];
  - Assim como na biologia, AGs selecionam características desejáveis por meio da aptidão [Luke, 2013];
  - Os valores de aptidão dos indivíduos determinam sua capacidade de sobreviver [Bozorg-Haddad et al., 2017].

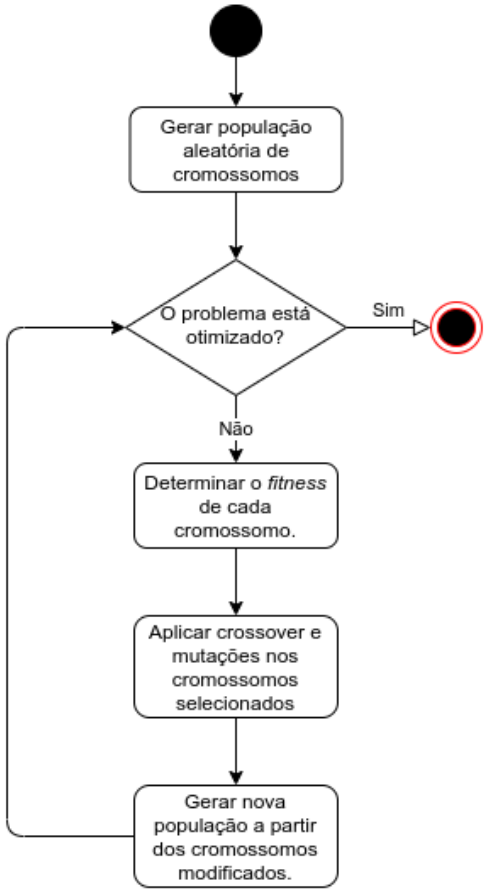
Conceitos

- AGs também possuem outros conceitos de inspiração biológica, como mutação, clonagem e *crossover*.
  - Mutação: operador que realiza a escolha aleatória de genes e estes têm seus valores são trocados pelos de seus genes alelos[5];
  - Crossover: operador que substitui os genes de um pai pelos genes correspondentes de outro para geração de um novo indivíduo (filho) [Coppin, 2004] [Luzia and Rodrigues, 2009].
- As características de um indivíduo partem da forma com se combinam os cromossomos dos pais [Luke, 2013].

[5] Genes alelos: são aqueles que ocupam o mesmo *lócus* em cromossomos homólogos e estão envolvidos na determinação de um mesmo caráter" [Magalhães, 2020]. Ex.: cor dos olhos - AA, Aa, aA, aa.

Método

- O algoritmo pode ser resumido no diagrama abaixo.



Fonte: Próprio autor

Método - Inicialização da população

INSTITUTO FEDERAL
Minas Gerais

- A inicialização, em si, é aleatória;
- Pontos principais desta fase:
  - Escolha do tamanho da população
    - População pequena não permite exploração do problema;
    - População grande causa perda de eficiência do método;
    - Não existe uma fórmula para definir o tamanho da população;
  - Definição do método que será aplicado à seleção dos indivíduos [Luzia and Rodrigues, 2009].

Método - Condição de término

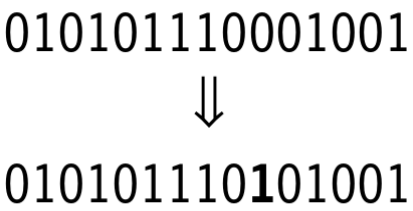
- Como o método é estocástico, devem ser definidos critérios de parada;
- Critérios de parada:
  - Limite de tempo;
  - Limite da quantidade soluções avaliadas;
  - Atingir uma determinada propriedade (erro, acurácia, etc) [Luzia and Rodrigues, 2009].

Método - Mutação e *Crossover*

- Estratégias possíveis:
  - Crossover-AND-Mutation: execução do *crossover* seguido de uma mutação, nesta ordem;
  - Crossover-OR-Mutation: possibilita a variação nas proporções entre os operadores ao longo da busca;
- Técnica recomendada por [Luzia and Rodrigues, 2009]:
  - No início do algoritmo é indicado utilizar uma alta taxa de *crossover* e aumentar gradualmente a taxa de mutação, de acordo com a convergência da população.

Método - Mutação

- Mutação: "operador unário[6] que realiza a escolha aleatória de um subconjunto de genes e seus valores são trocados pelos de seus genes alelos" [Luzia and Rodrigues, 2009] [Coppin, 2004]
  - A mutação é, em geral, feita de forma aleatória;
  - Um parâmetro define a taxa de mutação de cromossomos.



Fonte: [Coppin, 2004]

[6] Operador aplicado a um único argumento (gene) [Coppin, 2004].

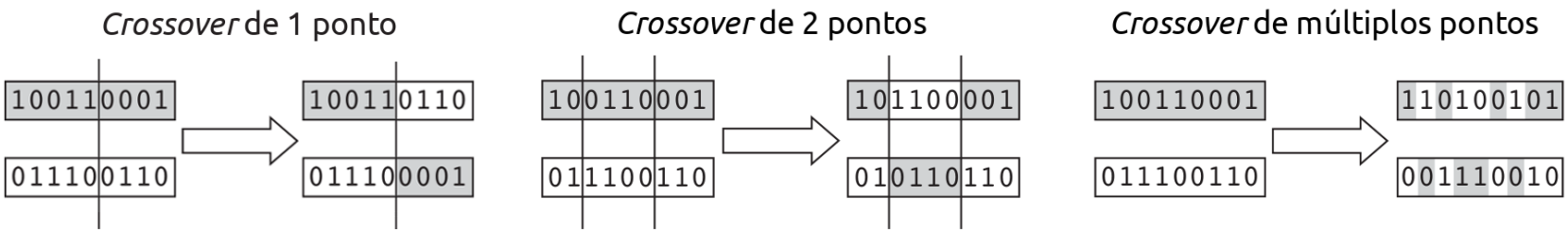
Método - *Crossover*

INSTITUTO FEDERAL
Minas Gerais

- Crossover: operador que substitui os genes de um pai pelos genes correspondentes de outro para geração de um novo indivíduo (filho) [Coppin, 2004] [Luzia and Rodrigues, 2009].
- Funcionamento do operação de *crossover*, para cromossomos de mesmo tamanho
  1. Seleção aleatória de um ponto de *crossover*;
  2. Quebra do cromossomo em duas partes, a partir do ponto de *crossover*;
  3. Recombinação dos cromossomos quebrados, combinando o início de um cromossomo com o final de outro [Coppin, 2004]..

Método - *Crossover*

- O *crossover* pode ser classificado quanto ao número de quebras (1 ponto, 2 pontos, n pontos);
- *Crossover* uniforme: uma probabilidade p é dada para determinar se um gene será substituído [Coppin, 2004].



Fonte: Adaptado de [Coppin, 2004]

Método - Seleção

- A cada geração, uma parte da população é selecionada, com base na função de avaliação, de modo a gerar indivíduos para uma próxima geração [Luzia and Rodrigues, 2009];
- Abordagens:
  - Avaliar todos os indivíduos;
  - Avaliar uma amostra aleatória de indivíduos;
    - *Tournament selection*: n indivíduos mais adaptados de cada torneio são selecionados para *crossover*;
    - *Roulette Wheel Selection*: seleção por roleta, como em um casino.

Vantagens e Desvantagens

- Vantagens:
  - Algoritmos muito eficientes na obtenção de soluções;
  - Possibilitam uma grande variedade de soluções;
  - Possuem capacidade análise de espaço de busca muito mais eficiente que algoritmos de busca local [Luzia and Rodrigues, 2009].
- Desvantagens:
  - O comportamento dos AGs é complexo e imprevisível, fugindo ao controle do desenvolvedor [Luzia and Rodrigues, 2009].
  - Alto custo computacional.

Outras Meta-Heurísticas

Busca Tabu
(Tabu Search)

Busca Tabu

- Estratégia de Busca Local proposta por Fred Glover em 1986;
- Armazena estados já visitados anteriormente de modo a evitar passar pelos mesmos caminhos [Coppin, 2004];
- Mantém uma lista de soluções candidatas visitadas (Lista Tabu) e somente visita-as novamente após um determinado intervalo de tempo [Luzia and Rodrigues, 2009].
  - A Lista Tabu é uma lista do tipo *short-term memory*, ou seja, os elementos são renovados com uma determinada frequência.

[Souza, 2011] indica que o trabalho de Hansen [Hansen, 1986], publicado em 1986 de forma independente do trabalho de Glover, pode ser considerado também Busca Tabu. Tal trabalho usa uma variação do Steepest Ascent Hill Climbing. [Luke, 2013] utiliza esse algoritmo para construção do pseudo-algoritmo da Busca Tabu.

Busca Tabu

- Quando uma solução máxima ou mínima é encontrada, o algoritmo força a busca em um ponto longe da solução atual [Luzia and Rodrigues, 2009];
- Não é permitido pelo algoritmo manter-se em um único local ou retornar ao ponto de solução anterior, forçando o algoritmo a percorrer novos caminhos [Luke, 2013];
- Com isso, o algoritmo consegue buscar em uma área maior do espaço de soluções.

Vantagens e Desvantagens

- Vantagens:
  - Capacidade de lidar com problemas combinatoriais difíceis;
  - Capacidade de lidar com restrições complexas [Luzia and Rodrigues, 2009].
- Desvantagens:
  - Se o espaço de busca for muito grande, a busca poderá ficar presa em uma mesma vizinhança;
  - O aumento da Lista Tabu pode reduzir esse problema, porém requer mais recursos;
  - Em problemas extremamente grandes, essa situação pode se agravar devido a limitações de tamanho da Lista Tabu [Luzia and Rodrigues, 2009]

Otimização por Colônia de Formigas (ACO)

Otimização por Colônia de Formigas

INSTITUTO FEDERAL
Minas Gerais

- Ant Colony Optimization (ACO) é um tipo de metaheurística construtiva inspirada na trilha de feromônios deixados por formigas, permitindo que outras encontrem alimento
  - Feromônio é um sinal químico liberado que dispara uma resposta natural em outros membros da espécie [Coppin, 2004] [Luzia and Rodrigues, 2009];
- O algoritmo ACO foi criado por Marco Dorigo em 1991 e publicado, posteriormente, em 1996 [Souza, 2011] [Bozorg-Haddad et al., 2017].

Otimização por Colônia de Formigas

INSTITUTO FEDERAL
Minas Gerais

- O algoritmo simula o comportamento de um conjunto de agentes simples (formigas artificiais) que cooperam para resolver um problema de otimização [Souza, 2011];
- As formigas são procedimentos estocásticos (a partir de eventos aleatórios) que constroem uma solução por meio da adição iterativa de componentes a uma solução parcial
  - A colônia move-se de forma concorrente e assíncrona construindo caminhos no espaço de busca [Luzia and Rodrigues, 2009].

Otimização por Colônia de Formigas

INSTITUTO FEDERAL
Minas Gerais

- São utilizados os seguintes procedimentos para adição de componentes à solução:
  1. Informações heurísticas sobre a instância do problema (se disponível);
  2. Trilhas de feromônio artificiais que mudam dinamicamente, em tempo de execução, para refletir a experiência de busca adquirida pelos agentes [Luzia and Rodrigues, 2009] [Souza, 2011].

Otimização por Colônia de Formigas

INSTITUTO FEDERAL
Minas Gerais

- O ACO é baseado na comunicação indireta, mediada por rastros de feromônios artificiais
  - Os rastros de feromônio funcionam como uma informação numérica distribuída;
  - As formigas artificiais utilizam esse rastro para construir, probabilisticamente, soluções para um dado problema;
  - Esses rastros refletem as experiências de busca das formigas durante a solução e podem ser utilizados como informação por outras formigas [Souza, 2011] [Luzia and Rodrigues, 2009].

Otimização por Colônia de Formigas

- A medida em que se movem, as formigas constroem novas soluções para o problema de otimização;
- Para evitar a convergência prematura para regiões subótimas, os algoritmos de ACO implementa uma ação denominada evaporação de feromônio
  - Este processo permite que rotas velhas desapareçam gradualmente, permitindo a manutenção da rota de melhor valor [Souza, 2011] [Luzia and Rodrigues, 2009].

Algoritmo

- O pseudo-algoritmo do ACO pode ser visto abaixo.

```
01: C ← {C1, ..., Cn} componentes
02: t ← número de trilhas para construir de uma só vez
03: f ← <f1,...,fn> feromônios dos componentes
04: Melhor ← nulo
05: repita
06:     P ← t trilhas, construídas por seleção iterativa de componentes baseada nos
feromônios e nas informações heurísticas
07:    para cada Pi em P faça
08:        se Melhor = nulo ou Qualidade(Pi) > Qualidade (Melhor) então
09:            Melhor ← Pi
10:     atualize f para os componentes baseado na qualidade para cada Pi em P em que
        eles participaram
11:     EvaporarFeromônio()
12:     AçõesDeSegundoPlano()
13: até que Melhor seja a solução ideal ou o tempo tenha se esgotado
14: devolva Melhor
```

Ant Colony Optimization

Fonte: [Luzia and Rodrigues, 2009]

O ACO não requer a implementação efetiva de formigas. Estas são usadas apenas para explicar a ideia do método. A implementação efetiva está relacionada apenas aos depósitos de ferômonios.

Vantagens e Desvantagens

- Vantagens:
  - A característica estocástica permite que as formigas construam uma grande variedade de soluções diferentes;
  - Flexibilidade: o ACO pode ser adaptado a diversas situações;
  - Capacidade de uso em problemas dinâmicos, cujos valores se alteram durante a ciclo de execução do programa [Luzia and Rodrigues, 2009].
- Desvantagens:
  - Possibilidade de estagnação;
  - Possibilidade de convergência prematura;
  - Alto custo computacional [Luzia and Rodrigues, 2009].

GRASP
Greedy Randomized Adaptive Search Procedures

GRASP

- O GRASP (*Greedy Randomized Adaptive Search Procedures*) é uma metaheurística utilizada para solução de problemas de otimização combinatória [Festa and Resende, 2002];
- Foi desenvolvido em 1989 por Feo e Resende [Souza, 2011] [Luzia and Rodrigues, 2009]

GRASP

INSTITUTO FEDERAL
Minas Gerais

- O GRASP é um processo iterativo, no qual cada iteração é constituída de 2 fases: [Festa and Resende, 2002]
  1. Construção: produz-se uma solução factível, de forma iterativa;
  2. Busca Local: busca-se um ótimo local na vizinhança das soluções construídas;

Fase 1 - Construção

- Produz-se uma solução factível, de forma iterativa
  - A cada iteração, é escolhido o próximo elemento a ser adicionado à solução;
  - Uma função avalia a lista de candidatos e verifica quais os benefícios gerados por cada um;
  - Os candidatos são ordenados em uma lista dos melhores candidatos, denominada *Restricted Candidate List*[7] (RCL);
  - O algoritmo escolhe aleatoriamente um elemento dessa lista [Festa and Resende, 2002] [Luzia and Rodrigues, 2009];

[7] [Souza, 2011] traduz o nome da lista para Lista Restrita de Candidatos (LRC).

Fase 2 - Busca Local

- A Fase de Construção, apesar de retornar boas soluções, não garante que a solução gerada seja um ótimo local [Festa and Resende, 2002];
- Com isso, uma segunda fase, de Busca Local, é utilizada para refinar a solução previamente retornada
  - Um método realiza busca de um máximo local na vizinhança dos resultados obtidos na fase anterior;
- O refinamento adequado da solução é baseado na escolha de um bom algoritmo de busca local para ser utilizado nesta fase.

Vantagens e Desvantagens

INSTITUTO FEDERAL
Minas Gerais

- Vantagens:
  - Implementação simples, facilmente paralelizável;
  - Flexibilidade: pode ser adaptado a diversas situações [Festa and Resende, 2002].
- Desvantagens:
  - A análise formal da qualidade das soluções geradas pelo GRASP é complexa;
  - Solução dependente da lista de candidatos (os resultados serão ruins se a lista for inadequada) [Festa and Resende, 2002].

Referências I

Belfiore, P. and Fávero, L. P. (2013).
Pesquisa operacional para cursos de engenharia.
Elsevier, 1 edition.

Bozorg-Haddad, O., Solgi, M., and Loáiciga, H. A. (2017).
Meta-heuristic and Evolutionary Algorithms for Engineering Optimization.
Wiley Series in Operations Research and Management Science. Wiley, 1 edition.

Coppin, B. (2004).
Artificial intelligence illuminated.
Jones and Bartlett illuminated series. Jones and Bartlett Publishers, 1 edition.

da Silva, D. M. (2014).
Inteligência Artificial - Slides de Aula.
IFMG - Instituto Federal de Minas Gerais, Campus Formiga.

da Silva, D. M. (2017).
Métodos Heurísticos - Simulated Annealing - Slides de Aula.
IFMG - Instituto Federal de Minas Gerais, Campus Formiga.

Dicas de Cálculo (2020).
Máximos e mínimos de uma função: teste da primeira derivada.
[Online]; acessado em 22 de Setembro de 2020. Disponível em: `https://www.dicasdecalculo.com.br/conteudos/derivadas/aplicacoes-de-derivadas/maximo-minimo-funcao/`.

Referências II

Festa, P. and Resende, M. G. (2002).
Grasp: An Annotated Bibliography, pages 325–367.
Springer US, Boston, MA.

Hansen, P. (1986).
The steepest ascent mildest descent heuristic for combinatorial programming.
In Congress on numerical methods in combinatorial optimization, Capri, Italy, pages 70–145.

Hoang Duong - Hoang Duong blog (2015).
Gradient descent and variants - convergence rate summary.
[Online]; acessado em 03 de Setembro de 2020. Disponível em:
http://hduongtrong.github.io/2015/11/23/coordinate-descent/.

Kirkpatrick, S., Gelatt, C., and Vecchi, M. (1983).
Optimization by simulated annealing.
In Fischler, M. A. and Firschein, O., editors, Readings in Computer Vision, pages 606–615. Morgan Kaufmann, San Francisco (CA).

Luke, S. (2013).
Essentials of Metaheuristics (Second Edition).
lulu.com, 2 edition.
[Online]; Disponível em: https://cs.gmu.edu/~sean/book/metaheuristics/.

Luzia, L. F. and Rodrigues, M. C. (2009).
Estudo sobre as metaheurísticas.
[Online]; acessado em 22 de Setembro de 2020. Disponível em:
https://www.ime.usp.br/~gold/cursos/2009/mac5758/LeandroMauricioHeuristica.pdf.

Referências III

Magalhães, L. (2020).
Genes alelos.
[Online]; acessado em 29 de Setembro de 2020. Disponível em:
https://www.todamateria.com.br/genes-alelos/.

Russel, S. and Norvig, P. (2013).
Inteligência artificial.
Campus - Elsevier, 3 edition.

Souza, M. J. F. (2011).
Inteligência computacional para otimização.
[Online]; acessado em 12 de Maio de 2021. Disponível em: http://www.decom.ufop.br/prof/marcone/Disciplinas/InteligenciaComputacional/InteligenciaComputacional.pdf.

Sucupira, I. R. (2004).
Métodos heurísticos genéricos: metaheurísticas e hiper-heurísticas.
PhD thesis, Universidade de São Paulo.
Acessado em 28 de Setembro de 2020. Disponível em:
https://www.ime.usp.br/~igorrs/monografias/metahiper.pdf.

Wikipedia contributors (2020).
Saddle point.
[Online]; acessado em 09 de Setembro de 2020. Disponível em:
https://en.wikipedia.org/wiki/Saddle_point.