INTELIGÊNCIA COMPUTACIONAL

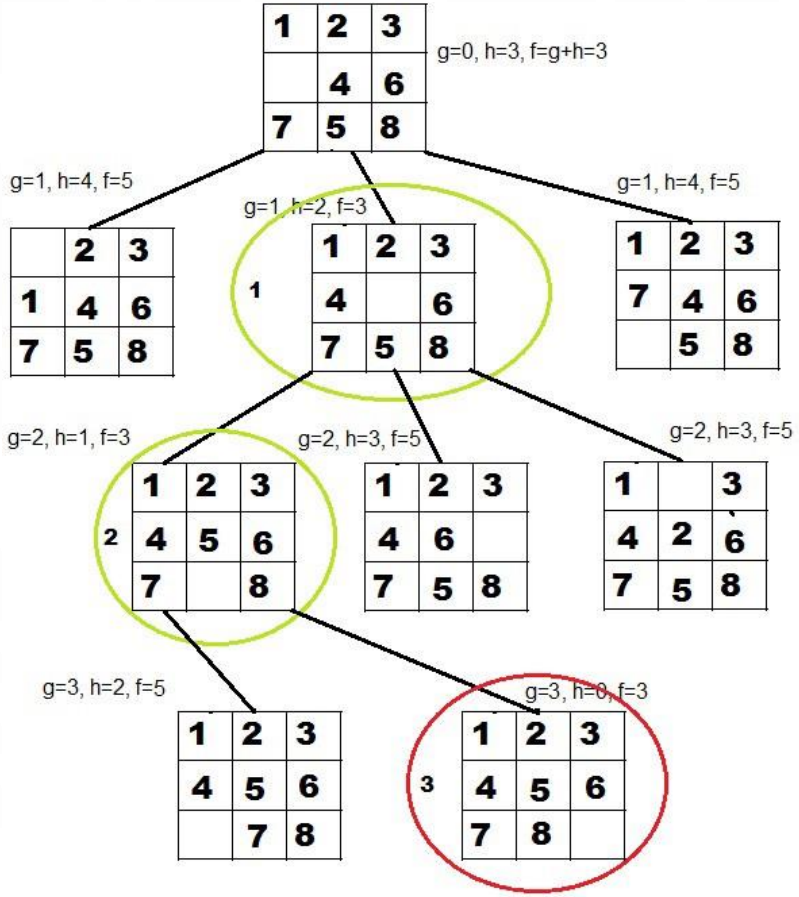
PROF. JOSENALDE OLIVEIRA

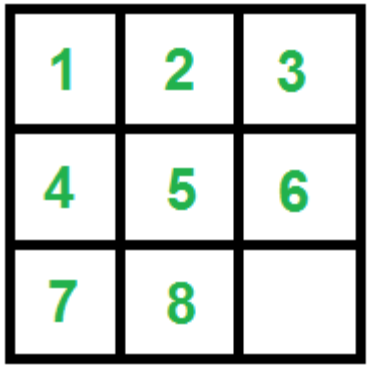
josenalde.oliveira@ufrn.br
https://github.com/josenalde/ic

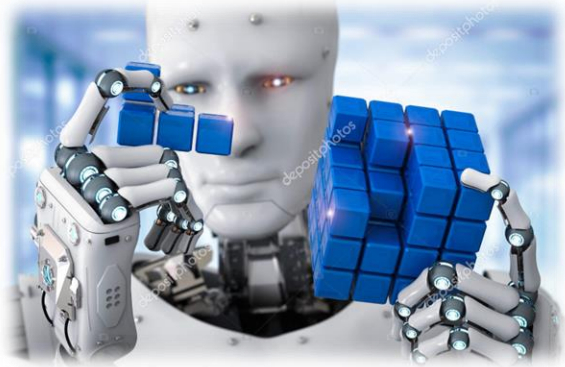
ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - UFRN

MAIS SOBRE HEURÍSTICAS

- **8-puzzle**



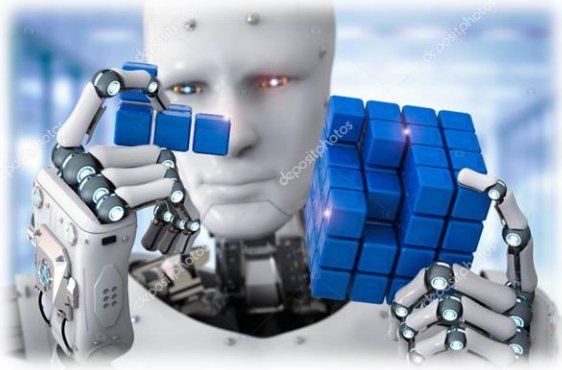




Heurística: g: quantidade de movimentos
h: peças “fora” do lugar em relação ao objetivo

MAIS SOBRE HEURÍSTICAS

- **2048**

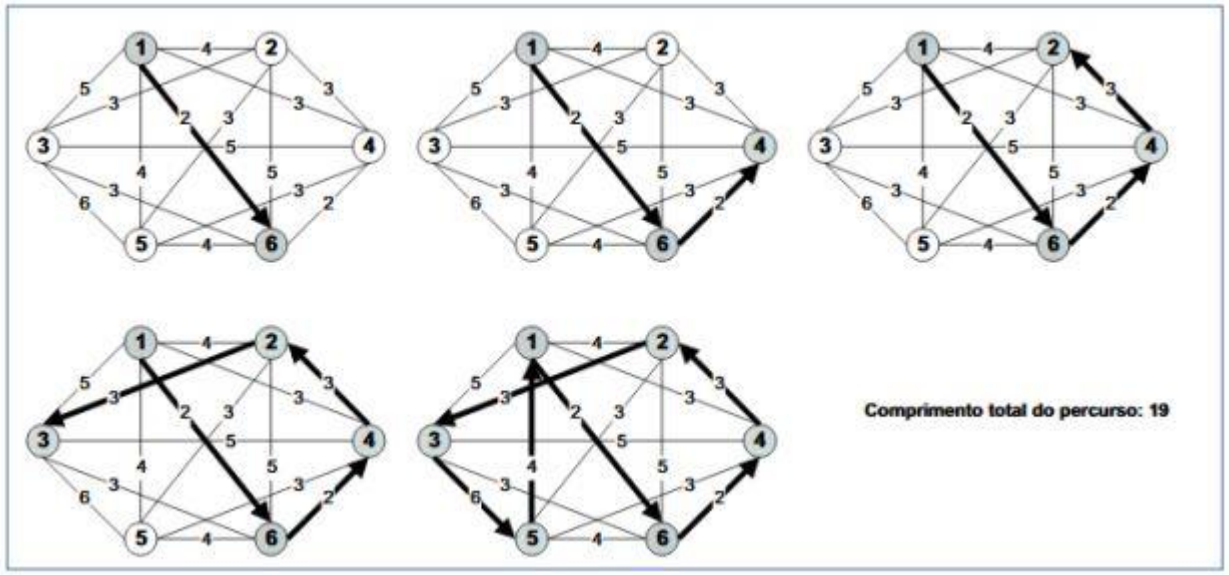


Discussão sobre heurísticas e implementação:

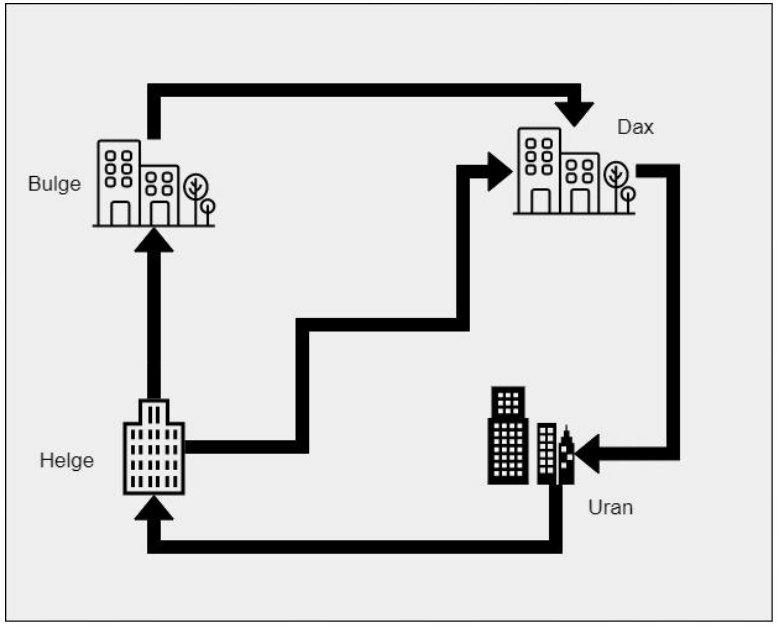
https://programming-techniques.com/ask-68-what-is-the-optimal-algorithm-for-the-game-2048/

MAIS SOBRE HEURÍSTICAS

- **E o problema do caixeiro viajante (TSP)?**



- **Distância entre as cidades e a ordem em que são visitadas**
- **Cada cidade visitada uma única vez**
- **Problema de otimização**
- **Logística (Ifood, Uber etc.), fabricação, sequenciamento DNA etc.**



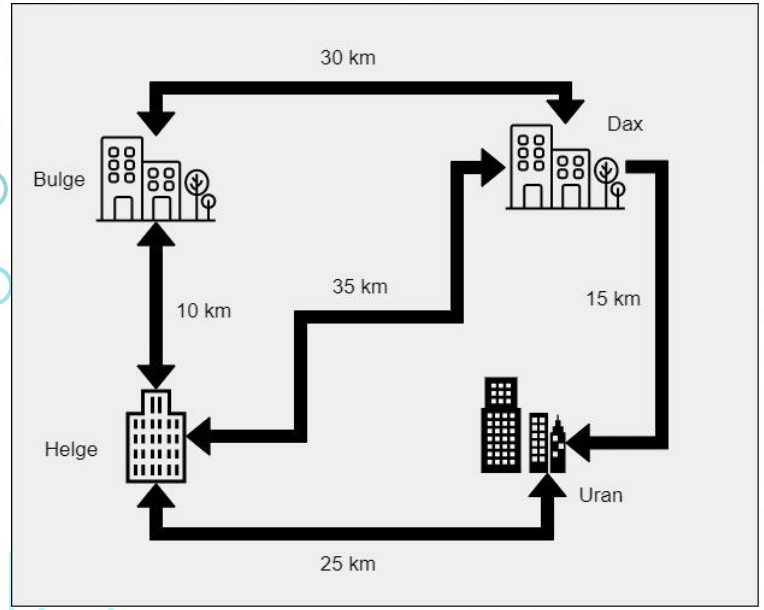
Rota válida: Dax-Uran-Helge-Bulge
Rota inválida: Helge-Dax-Uran-Helge-Bulge

MAIS SOBRE HEURÍSTICAS

- **E o problema do caixeiro viajante (TSP)?**

- **Por força bruta testa todas as combinações e ver a melhor!**
  - **Segue de depois avalia! Ruim!**
- **Com heurística, pensa antes de ir!**

Exemplo: Se eu estou em uma cidade, vou para próxima cidade que possui o menor custo de caminhada



PROBLEMAS COMBINATORIAIS:
Variáveis de decisão inteiras ou binárias, onde as soluções são combinações factíveis dos dados de entrada. À medida que n cresce, o tempo cresce exponencialmente

Cidades	Combinações possíveis
1	1
2	2
3	6
4	24
5	120
6	720
7	5040
8	40320
9	362880
10	3628800
11	39916800
12	479001600
13	6227020800
14	87178291200
15	1307674368000
16	20922789888000
17	355687428096000

!

MAIS SOBRE HEURÍSTICAS

- **E o problema do caixeiro viajante (TSP)?**

Formulação padrão (1960)

Miller-Tucker-Zemlin

$$\min Z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}$$

$$\text{sa} \quad \sum_{i=1}^{n} x_{ij} = 1, \qquad j = 1, 2, \ldots, n$$

$$\sum_{j=1}^{n} x_{ij} = 1, \qquad i = 1, 2, \ldots, n$$

$$u_i - u_j + nx_{ij} \le n - 1 \qquad \forall 2 \le i \ne j \le n$$

$$x_{ij} \in \{0,1\} \qquad \forall i, j$$

Para dimensões elevadas, a resolução por métodos de programação matemática é proibitiva em termos de tempos computacionais. Portanto, a abordagem de solução heurística e de aproximação é mais útil para as aplicações que preferem o tempo de execução do algoritmo em relação à precisão do resultado

Exemplos de heurísticas

Heurísticas para resolver o problema do caixeiro viajante são métodos por meio dos quais a solução ótima para o TSP é procurada. Embora as heurísticas não possam garantir que a solução ótima seja encontrada, ou não ao menos em tempo real, um grande número delas encontrará uma solução próxima à ótima, ou mesmo encontrará uma solução ótima para certos casos do problema do caixeiro viajante.

Para avaliar o desempenho de um algoritmo, podemos verificar como o tempo de execução dele cresce conforme aumentamos o volume de dados oferecido como entrada. Um problema é intratável se o tempo necessário para resolvê-lo é considerado inaceitável para os requisitos do usuário. O problema do caixeiro viajante é um exemplo de problema intratável por algoritmos convencionais em uma busca exaustiva

MAIS SOBRE HEURÍSTICAS

- **Heurísticas construtivas e de melhoria/refinamento**

- ***Construtiva:*** Usando heurística construtiva constrói-se uma solução passo a passo (em geral do zero) segundo um conjunto de regras pré-estabelecido. Estas regras estão relacionadas com: a escolha do ciclo ou nó inicial da solução – inicialização; um critério de escolha do próximo elemento a juntar à solução – seleção; a escolha da posição onde esse novo elemento será inserido – inserção. Na construtiva Gulosa, a cada passo é adicionado um único elemento, sendo este o melhor segundo a função objetivo/função de avaliação. O método termina quando todos os elementos forem inseridos na solução

- ***Refinamento*:** inicia com uma solução (viável) e tenta melhorar a solução, através de modificações nos elementos da solução corrente

MAIS SOBRE HEURÍSTICAS

- **Para o TSP, exemplo de heurísticas construtivas**
  - *Inserção do vizinho mais próximo*
  - *Inserção mais barata*
  - *Algoritmo de Christofides*
  - *Método das economias (savings)*
  - *Método convex hull*
- ***Para o TSP, exemplo de heurísticas de melhoria***
  - *2-opt, 3-opt, etc.*

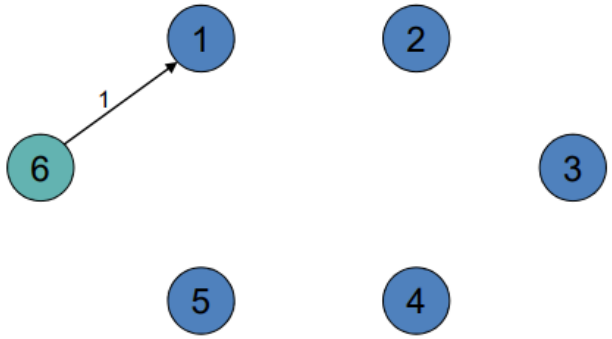
MAIS SOBRE HEURÍSTICAS

- **Inserção do vizinho mais próximo:** Construir uma rota adicionando, a cada passo, a cidade mais próxima da última cidade inserida e que ainda não foi visitada

1)

Cid.	1	2	3	4	5	6
1	0	2	1	4	9	1
2	2	0	5	9	7	2
3	1	5	0	3	8	6
4	4	9	3	0	2	6
5	9	7	8	2	0	2
6	1	2	6	6	2	0

i	j	d_{ij}
6	1	1
6	2	2
6	3	6
6	4	6
6	5	2

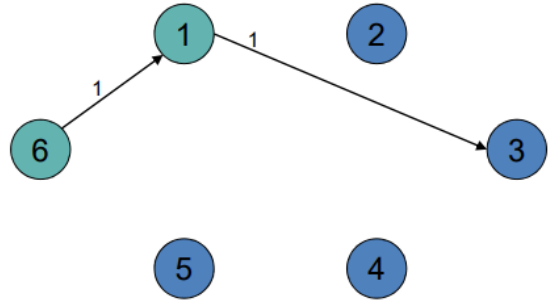


Distância total: 1

2)

Cid.	1	2	3	4	5	6
1	0	2	1	4	9	1
2	2	0	5	9	7	2
3	1	5	0	3	8	6
4	4	9	3	0	2	6
5	9	7	8	2	0	2
6	1	2	6	6	2	0

i	j	d_{ij}
1	2	2
1	3	1
1	4	4
1	5	9



Distância total: 2

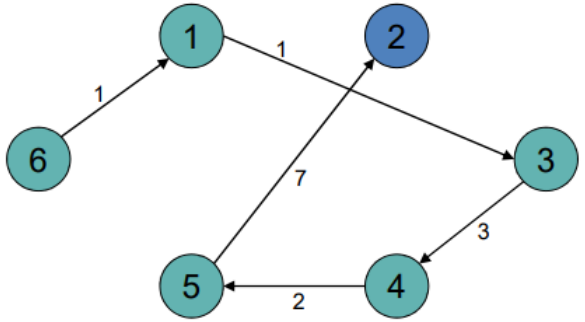
MAIS SOBRE HEURÍSTICAS

- **Inserção do vizinho mais próximo:** Construir uma rota adicionando, a cada passo, a cidade mais próxima da última cidade inserida e que ainda não foi visitada

5)

Cid.	1	2	3	4	5	6
1	0	2	1	4	9	1
2	2	0	5	9	7	2
3	1	5	0	3	8	6
4	4	9	3	0	2	6
5	9	7	8	2	0	2
6	1	2	6	6	2	0

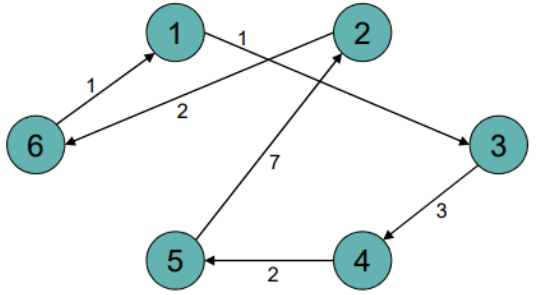
i	j	d_{ij}
5	2	7



Distância total: 7+7=14

6)

Cid.	1	2	3	4	5	6
1	0	2	1	4	9	1
2	2	0	5	9	7	2
3	1	5	0	3	8	6
4	4	9	3	0	2	6
5	9	7	8	2	0	2
6	1	2	6	6	2	0

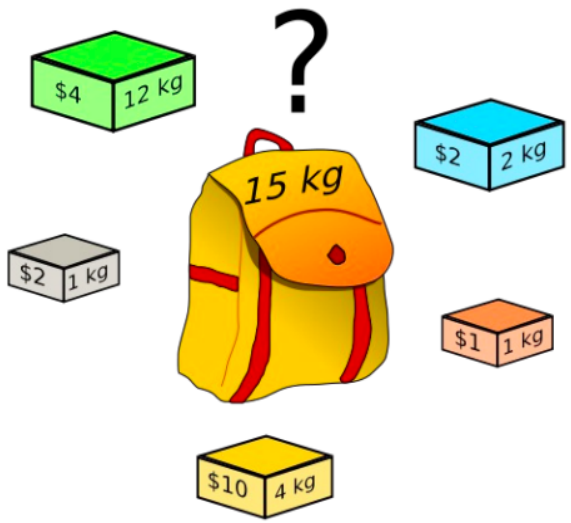


Distância total: 14+2=16

https://github.com/shunji-umetani/tsp-solver

MAIS SOBRE HEURÍSTICAS

- **Problema da mochila**



- Vários itens que gostaria de levar em uma mochila
- Cada item com um peso e um benefício (valor)
- Há uma capacidade limite de peso
- Deve-se carregar itens com o máximo valor total sem superar o limite de peso

MAIS SOBRE HEURÍSTICAS

- **Problema da mochila – exemplo de heurística**

Seja, então, uma mochila de capacidade $b = 23$ e os 5 objetos da tabela abaixo, com os respectivos pesos e benefícios.

Objeto (j)	1	2	3	4	5
Peso (w_j)	4	5	7	9	6
Benefício (p_j)	2	2	3	4	4

Construamos uma solução para esse problema usando a seguinte idéia: adicionemos à mochila a cada passo, o objeto mais valioso por unidade de peso e que não ultrapasse a capacidade da mochila. Reordenando os objetos de acordo com a relação p_j/w_j, obtemos:

Objeto (j)	5	1	4	3	2
Peso (w_j)	6	4	9	7	5
Benefício (p_j)	4	2	4	3	2
(p_j/w_j)	0.67	0.50	0.44	0.43	0.40

Representemos uma solução s por um vetor binário de n posições.

MAIS SOBRE HEURÍSTICAS

- **Problema da mochila – exemplo de heurística**

Passo 1 : Adicionemos, primeiramente, o objeto 5, que tem a maior relação p_j/w_j

$s = (00001)^t$

$f(s) = 4$

Peso corrente da mochila $= 6 <$ b $= 23$

Passo 2 : Adicionemos, agora, o objeto 1, que tem a segunda maior relação p_j/w_j

$s = (10001)^t$

$f(s) = 6$

Peso corrente da mochila $= 10 <$ b $= 23$

Passo 3 : Adicionemos, agora, o objeto 4, que tem a terceira maior relação p_j/w_j

$s = (10011)^t$

$f(s) = 10$

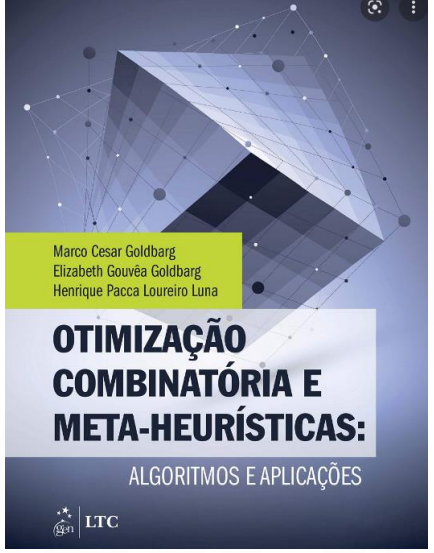
Peso corrente da mochila $= 19 <$ b $= 23$

Passo 4 : O objeto a ser alocado agora seria o terceiro. No entanto, esta alocação faria superar a capacidade da mochila. Neste caso, devemos tentar alocar o próximo objeto com a maior relação p_j/w_j, que é o objeto **2**. Como também a alocação desse objeto faria superar a capacidade da mochila e não há mais objetos candidatos, concluímos que a solução anterior é a solução final, isto é: $s^* = (10011)^t$ com $f(s^*) = 10$.

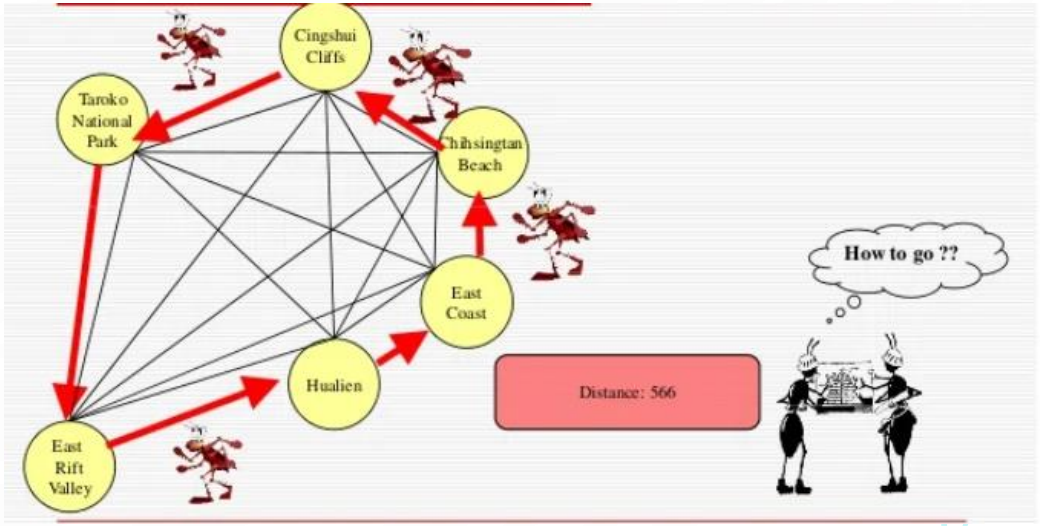
f(s): benefício

https://rosettacode.org/wiki/Knapsack_problem/Unbounded

META HEURÍSTICAS

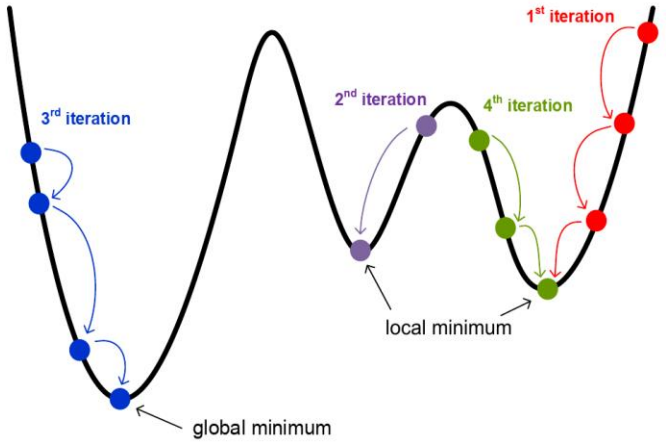


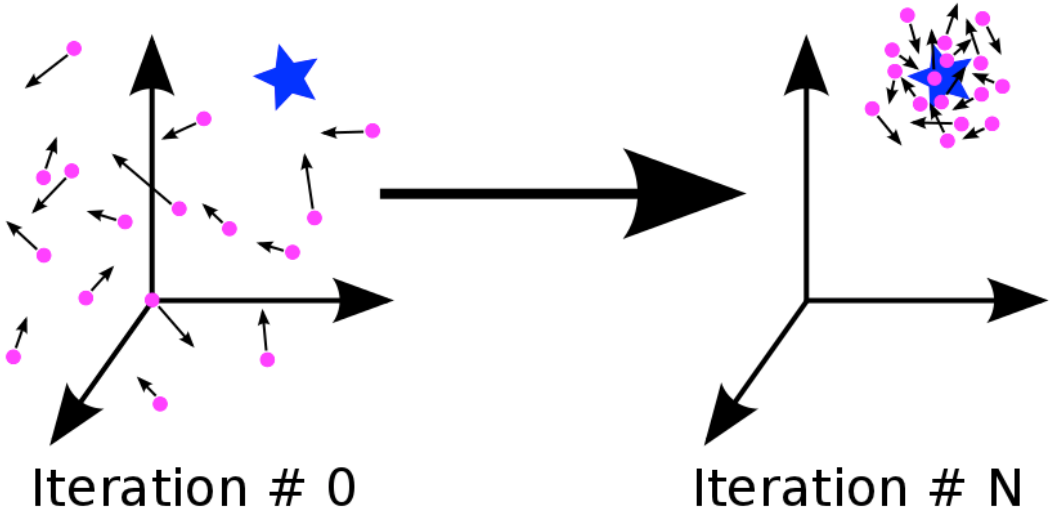
- Uma meta-heurística **é um método heurístico para resolver de forma genérica problemas de otimização** (normalmente da área de otimização combinatória). Meta-heurísticas são geralmente aplicadas a problemas para os quais não se conhece algoritmo eficiente (veja problemas NP-completos).
- As **metaheurísticas são** procedimentos que guiam outras **heurísticas**, ou seja, procedimentos computacionais, usualmente de busca local, explorando o espaço de soluções além do ótimo local. As **metaheurísticas** consideram boas características das soluções encontradas para explorar novas regiões promissoras.



META HEURÍSTICAS

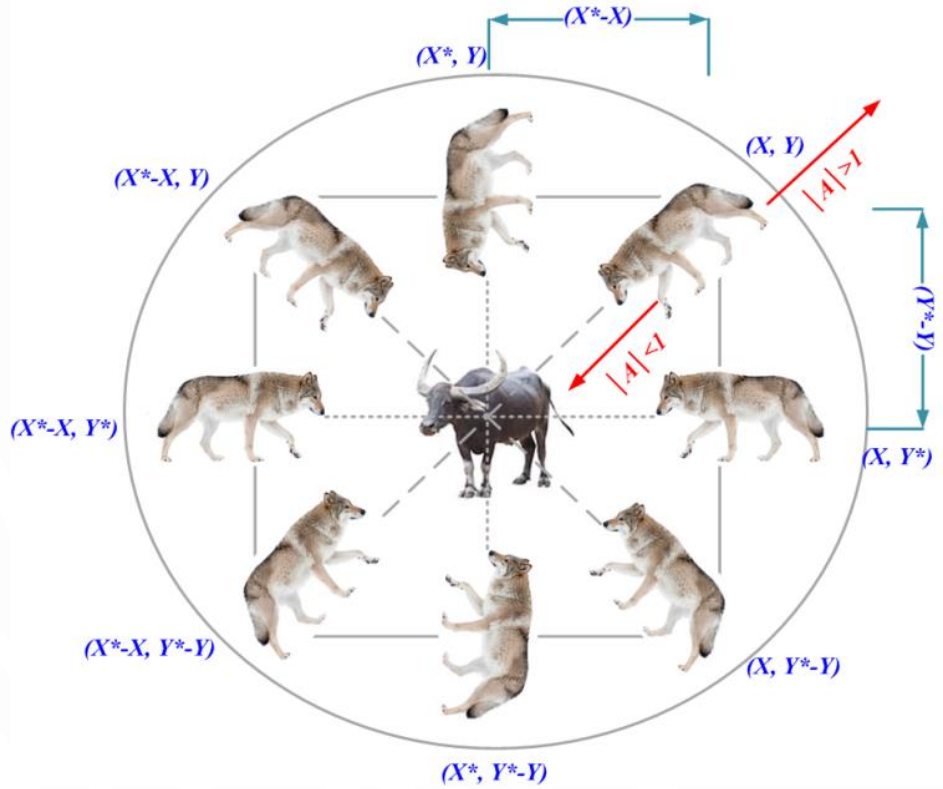
- Problemas com funções objetivo genéricas, não lineares, bem como restrições genéricas

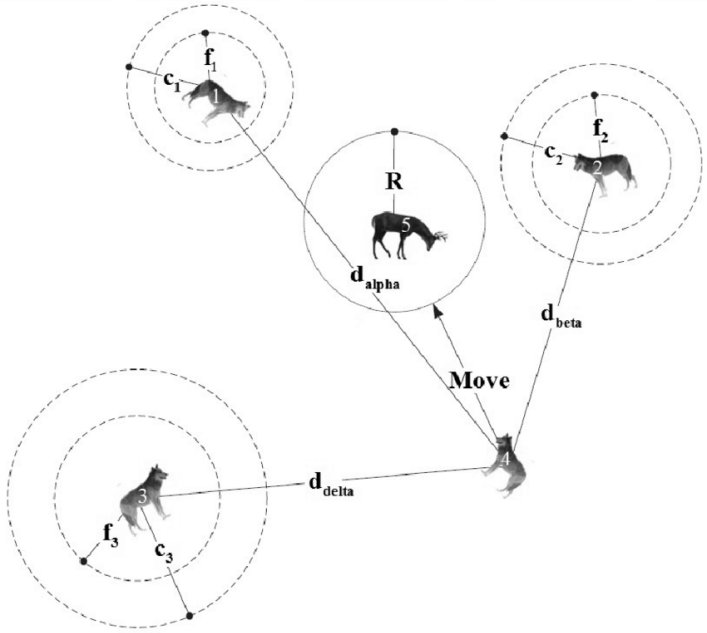




Meta-heurísticas evolutivas: lidam com uma população de soluções, que evolui, principalmente, através da interação entre seus elementos.

META HEURÍSTICAS

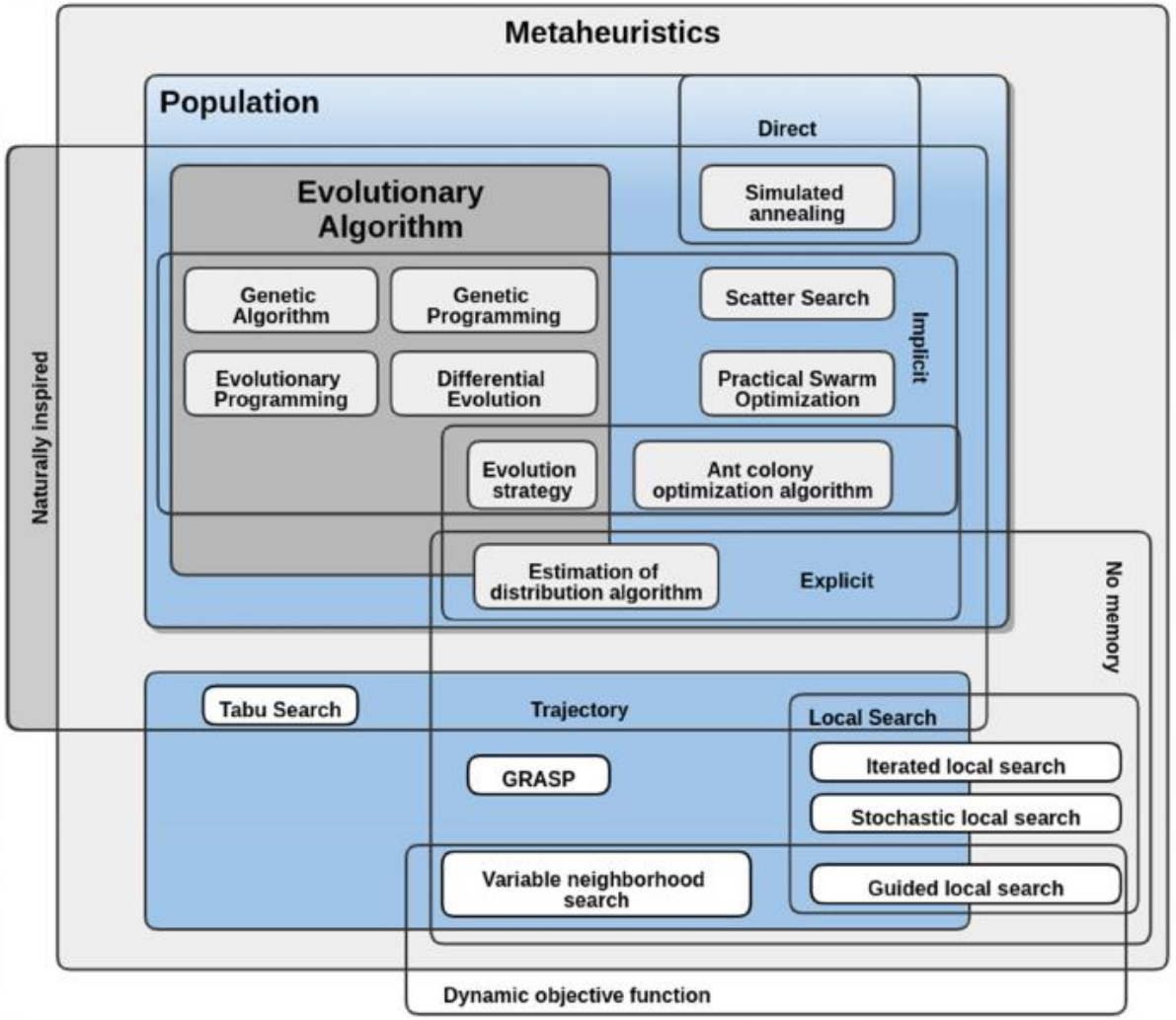




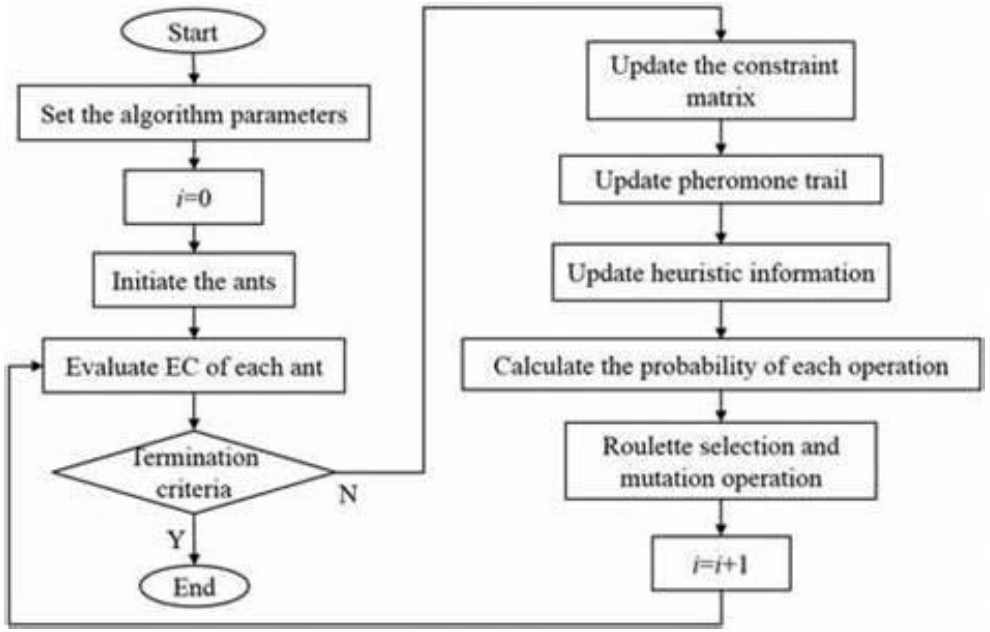
META HEURÍSTICAS – EXEMPLO DE APLICAÇÃO

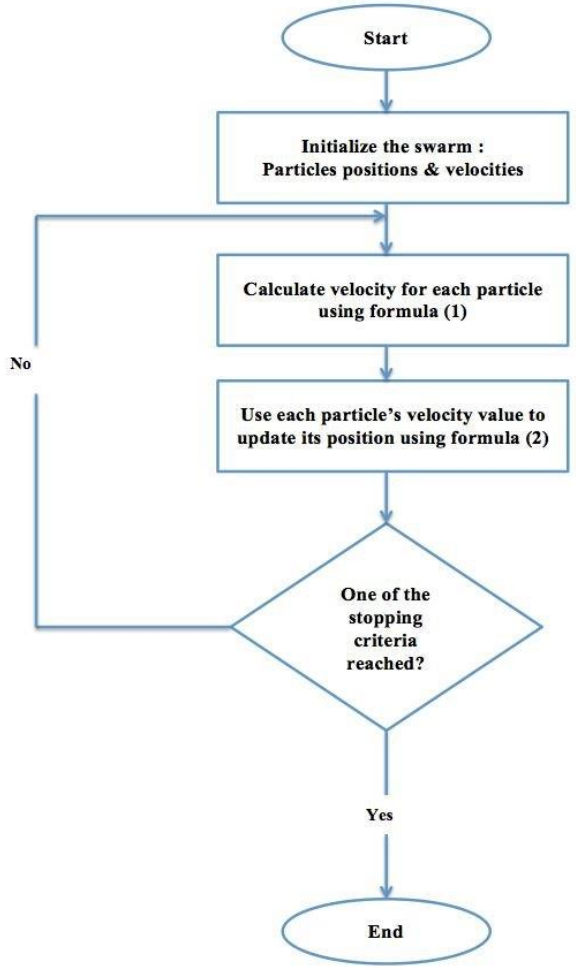
- Nos últimos anos, a técnica agrícola denominada Rotação de Culturas tem se destacado devido a sua importância econômica, ambiental e social e, uma vez empregada pelos produtores pode trazer-lhes muitos benefícios. São apresentadas metodologias para determinar planejamentos de plantio que promovam a **maximização da lucratividade da área cultivada**, levando em conta o **atendimento à necessidade de adubação verde, o período de pousio, as restrições de plantio para lotes vizinhos e para sequência de culturas**. Nesta abordagem, inicialmente é adaptado um ***modelo matemático para planejar a rotação de culturas*** e em seguida são discutidas estratégias de resolução deste. Trata-se de um problema de natureza combinatorial e **envolve elevado número de variáveis e de restrições**, razão pela qual o uso de heurísticas é adequado. Assim, são apresentadas as metaheurísticas: **algoritmo genético, simulated annealing e híbridas**, as quais apresentam poucas exigências computacionais. Os resultados mostram que o modelo é aplicável na prática e que os métodos propostos são robustos e eficientes, podendo ser usados no apoio aos produtores na tomada de decisão com objetivos econômico e ambiental.

META HEURÍSTICAS – EXEMPLO DE APLICAÇÃO



META HEURÍSTICAS – EXEMPLO DE APLICAÇÃO





1. Gerar população inicial (aleatória ou com 'entrevista')
2. Avaliar sua 'adequação'
3. Alterar características da população (algum elitismo?)
4. Verificar se atende critério
5. Retornar ao passo 2

META HEURÍSTICAS – EXEMPLO DE APLICAÇÃO

https://seyedalimirjalili.com/gwo