

Inteligência Artificial

E essa tal de inteligência artificial ?

É ficção científica ?

Qual o uso prático disso ?

Inteligência Artificial

- Não, não é ficção!
- É uma área com muita pesquisa e bem fundamentada, inclusive com muitas aplicações na indústria;
- Principal motivador hoje: Big Data;
- Trata-se do desenvolvimento de robôs? (mídia);



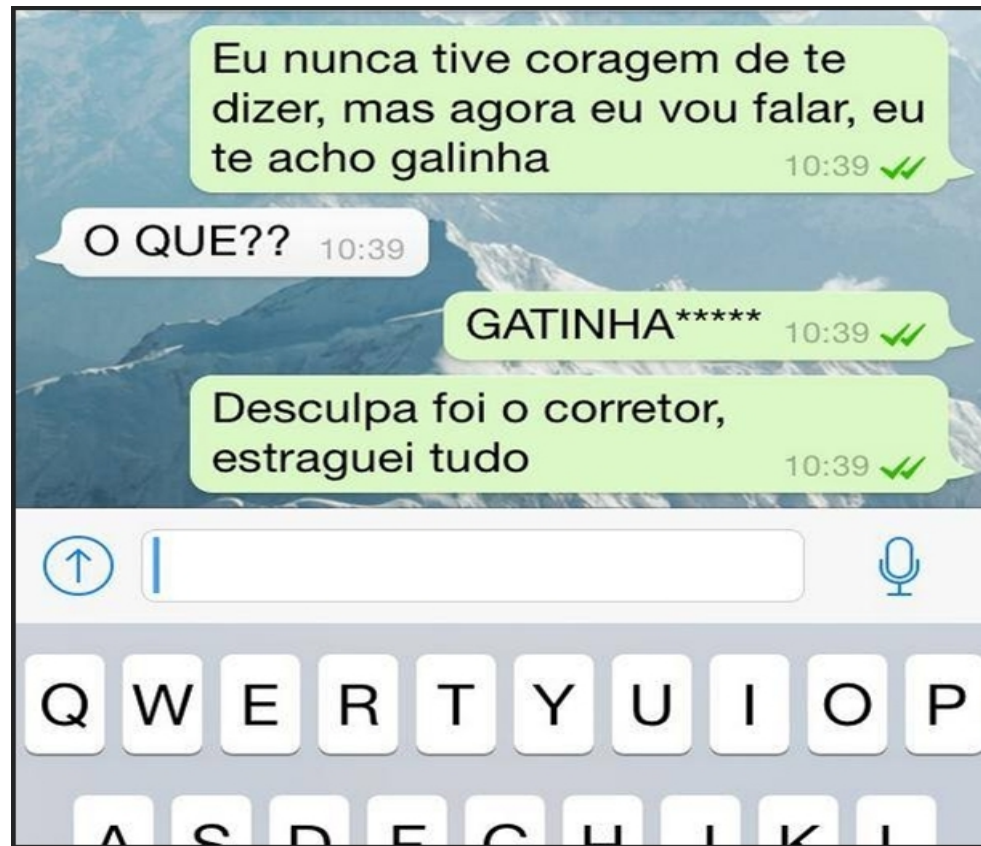
Exemplos industriais

- Sistemas de recomendação do Netflix:
<http://www.businessinsider.com/netflix-using-ai-to-suggest-better-films-2014-2>



Exemplos industriais:

- Corretor ortográfico do WhatsApp:



O que é inteligência? (não tem definição fácil)



O que é inteligência artificial?

- Capacidade mental de:
 - Planejar;
 - Raciocinar;
 - Resolver problemas;
 - Aprender;
 - Compreender ideias;
 - Etc.
- Na etimologia, o termo inteligência refere-se a quem sabe escolher: selecionar/escolher as melhores opções;
- A inteligência artificial tem como objetivo desenvolver mecanismos que consigam simular o raciocínio humano, ou seja, a sua inteligência.

O que é inteligência artificial?

- Através de técnicas de inteligência artificial, podemos ensinar um robô caminhar:
- Machine learning (aprendizado de máquina):
<https://www.youtube.com/watch?v=SBf5-eF-Elw>

Subáreas da Inteligência Artificial

- Sistemas especialistas – exemplo: simulação de experts em uma determinada área de conhecimento;
- Aprendizado de máquina (machine learning) – exemplo: redes neurais;
- Agentes inteligentes;
- Processamento de linguagem natural (PLN) – estuda os problemas da geração e compreensão da linguagem;
- etc.

O ideal humano

- **Fidelidade ao desempenho humano** são sistemas que pensam e/ou como agem os seres humanos.
- Máquinas que executam funções que exigem inteligência quando feitas por humanos;
- Fazer tarefas que hoje são melhor desempenhadas por humanos.

Agente inteligente

- O agente é uma entidade computacional que funciona de forma autônoma;
- Um sistema pode ser visto como um agente se for capaz de perceber seu ambiente *adaptando-se* a mudanças e tendo a capacidade de assumir metas.

Teste de Turing (1950)

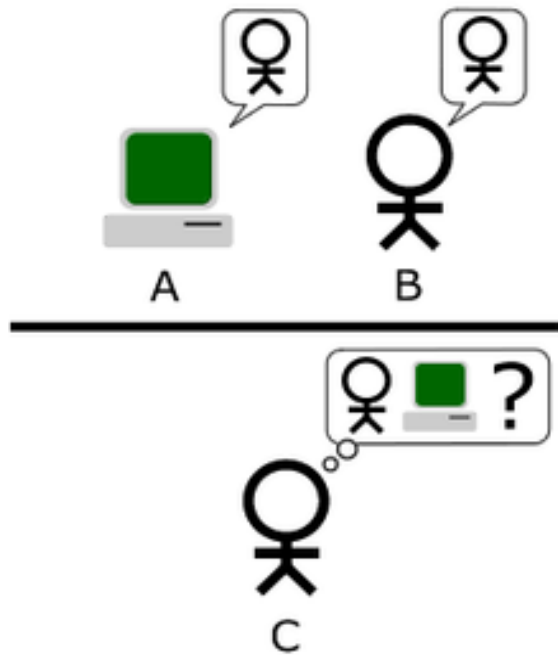
- O Teste de Turing testa a capacidade de uma máquina exibir **comportamento inteligente equivalente a um ser humano**, ou indistinguível deste:

“Se você não consegue distinguir se é uma máquina é realmente uma máquina ou é um humano, a máquina passa no Teste de Turing”;

- Programar um computador para passar neste teste é **muito** difícil.

Teste de Turing

https://pt.wikipedia.org/wiki/Alan_Turing



Filme: O jogo da imitação

Teste de Turing (1950)

- Esse teste foi proposto para fornecer uma definição operacional de inteligência;
- O computador passará no teste se um interrogador humano, depois de propor algumas perguntas, **não** for capaz de distingui-lo de um ser humano.

https://www.youtube.com/watch?v=FO0X_Cdevmo

Teste de Turing (1950)

Algumas capacidades exigidas para tentar passar no teste:

- Processamento de Linguagem Natural;
- Representação de conhecimento;
- Raciocínio Automatizado;
- Aprendizado de máquina.

Inteligência Artificial

- IA é o ramo da Ciência da Computação que lida com automação do pensamento e comportamento inteligente;
- Na prática, pesquisadores focam em alguma característica particular da inteligência;
- Constroem sistemas para auxiliar os humanos na solução de problemas complexos...

Algumas questões envolvendo IA

- Como o cérebro processa informações?
- Como construir computadores e algoritmos eficientes?
- Como a linguagem se relaciona com o comportamento?
- Como tomar decisões para maximizar uma recompensa?
- Como seres humanos e animais pensam e agem?

Um pouco de história...

- SAINT (1963): resolver problemas de cálculo integral;
- STUDENT (1967): resolver problemas clássicos de álgebra;
- ANALOGY (1968): resolver problemas de analogia geométrica, aqueles mesmos problemas que aparecem em testes de QI.

Um pouco de história...

Até que em 1996, computadores se tornam campeões de xadrez (Deep Blue)...



Projeto Watson da IBM venceu humanos em um jogo de perguntas e respostas em 2011.

Exemplo de “Entendimento da linguagem natural”



Aplicações

- A IA se torna ciência, através da adoção de métodos científicos, menos tentativa-e-erro, ganhando mais formalismo e uso prático;
- Existem muitos domínios de aplicação da IA:
Aviação, bioinformática, biometria, química, medicina, telecomunicações, detecção de fraudes, marketing, etc...

- Exemplo robôs humanoides: reconhecimento de padrões de som e vídeo, processando saídas em falas e expressões.

https://www.youtube.com/watch?v=m5ikbbe_-fl

- Exemplo carro autônomo.

<https://www.youtube.com/watch?v=TsaES--OTzM>

- Robocup: www.robocup2015.org

Questões éticas...

- As pessoas podem perder seus empregos?
- Sistemas de IA podem ser utilizados para fins indesejáveis?
- Pode haver perda de responsabilidade?
Exemplo: diagnóstico médico.

Sistemas especialistas

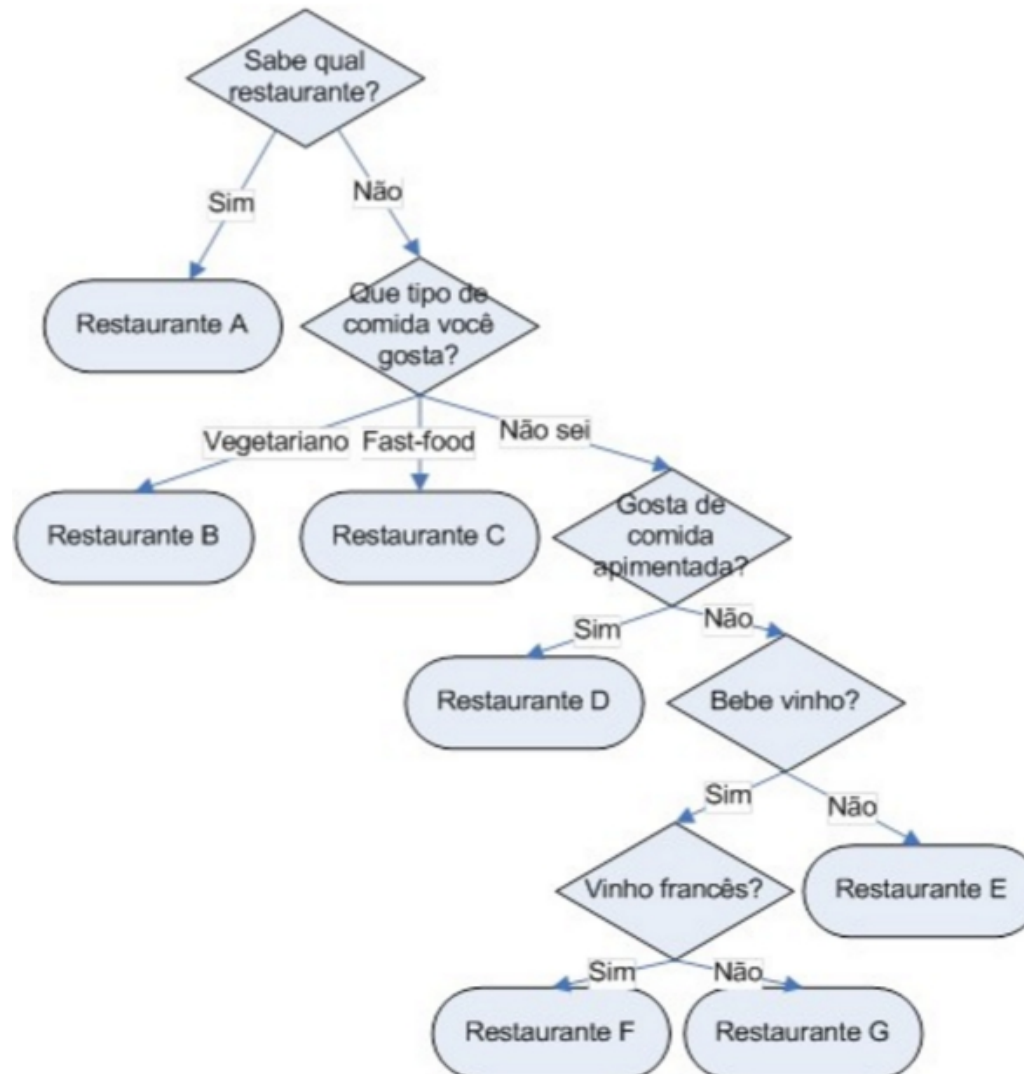
- São sistemas destinados a solucionar problemas em campos **específicos** de conhecimento. Estes devem ter desempenho comparável ao dos **especialistas** humanos na execução dessas tarefas.

Componentes

- Base de conhecimentos composta de fatos e regras;
- Mecanismo de inferência;
- Interface com o usuário.

Exemplo de SE:

Como chegar a um restaurante?



T1:

- Pesquisar sobre o MYCIN;
- Escolher uma doença e montar um SE simples de identificação;
- Dupla;
- Entregar **próxima aula**;

Resolução de problemas por busca

Busca é uma das mais poderosas abordagens para a resolução de problemas em IA.

Objetivo da busca: encontrar determinado objeto em uma coleção.

Resolução de problemas por busca

A busca explora sistematicamente as alternativas encontrando a sequência de passos para uma solução.

Ex.: quero viajar de uma cidade até outra, através da menor rota.

- Qual o objetivo (*estado*) do exemplo?
- Qual é a *medida* de desempenho?
- Quais são os *estados intermediários* do exemplo para chegar no objetivo?

Busca-se uma **sequência de ações/operadores** que leve a *estados desejáveis* (objetivos).

- O *estado* pode ser visto como uma estrutura de dados;
- **Estrutura de dados** é uma forma de organizar os dados.

Conceitos

- Três etapas da resolução dos problemas de busca: formular, buscar e executar;
- Teste de objetivo: determina se um dado estado é um estado objetivo;
- Função de custo de caminho: atribui um custo a cada caminho de uma estrutura (exemplo: grafo).

Conceitos

- Caminho: sequência de estados conectados;
- Custo de caminho: medida de desempenho;
- Para encontrar o menor caminho de um terminal para outro, uma solução para esse problema seria um caminho desde o estado inicial até o estado objetivo;
- A solução ótima tem o **menor custo de caminho** entre todas as possíveis soluções.

Conceitos: algoritmo básico

A estratégia de busca que determina qual estado será visitado e expandido;

A **fronteira** nada mais é do que os conjuntos a serem visitados;

Algoritmo:

- Selecione (e remove) o primeiro estado da fronteira do espaço de estados;
- Se a fronteira está vazia, então o algoritmo termina, ou seja, a busca termina com falha.

Conceitos

- Se o estado atual for um estado objetivo?
- Se for, então retorna o estado, a busca termina com sucesso;
- Se o estado atual não for o estado objetivo, então é gerado um novo conjunto de estados pela aplicação de operadores ao estado selecionado.

Algoritmo genérico de busca com fronteira:

1) Seleciona (e remove) o primeiro estado da fronteira do espaço de estados.

* se a fronteira está vazia, então o algoritmo falha.

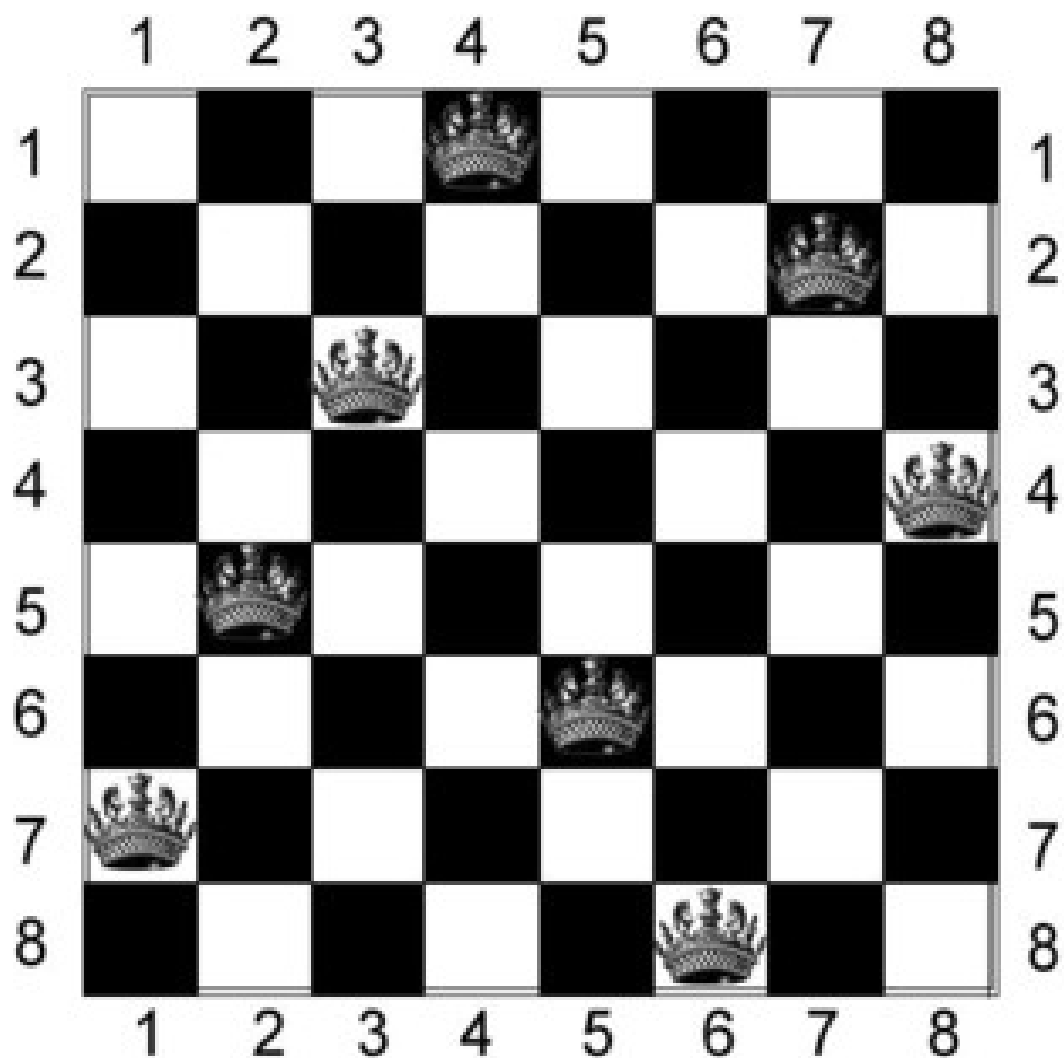
2) Testa se o estado é um estado objetivo.

* se sim,então retorna o estado (sucesso).

3) Gera um novo conjunto de estados;

4) Insere os estados gerados na fronteira de acordo com a **estratégia de busca** e volta para o passo 1.

O problema das 8 rainhas...



Qual é o objetivo?

O objetivo é posicionar as 8 rainhas em um tabuleiro de xadrez de forma que nenhuma das rainhas se ataquem por linha, coluna ou diagonal.

O problema das 8 rainhas...

E se aumentarmos o número de rainhas? O que acontece?

Para 100 rainhas podemos ter 10^{400} estados!!

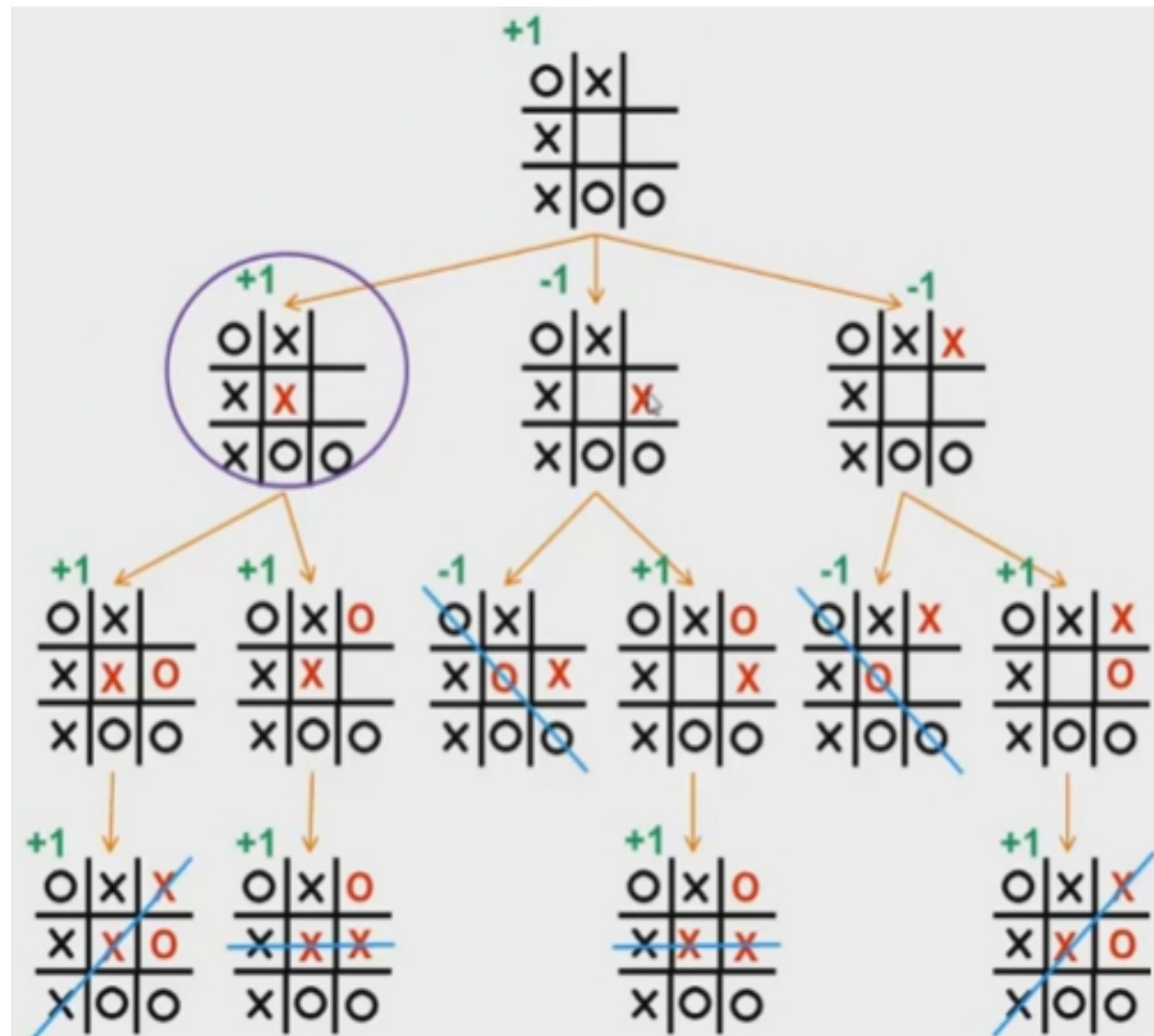
Exemplo: sistema de planejamento de viagens aéreas

- Cada estado poderia ser uma posição (aeroporto) e a hora atual;
- Estado inicial: especificado por cada problema;
- Função para gerar novos estados: retorna um estado após tomar algum voo programado;
- Teste de objetivo: estar no destino após um tempo especificado;
- Custo de caminho: tempo de viagem.

Problema do caixeiro-viajante: consiste em visitar um conjunto de cidades (vértices) **uma única vez** usando o percurso mais curto;

<https://www.youtube.com/watch?v=SC5CX8drAtU>

Exemplo: Algoritmo MinMax no Jogo da Velha



Estratégias de busca

- Diferentes estratégias se distinguem pela ordem em que os nós são expandidos.
- Existem estratégias que sabem se um estado é mais promissor que outro. São chamadas estratégias com **informação** ou **busca heurística**;
- Na **busca sem informação** (cega) a informação sobre os estados são somente aquelas fornecidas na definição do problema;
 - Somente geram sucessores e distinguem se um estado é objetivo ou não;

Estratégias de busca

- As estratégias determinam a ordem de ramificação dos nós. Exemplos:
 - a) Busca em largura;
 - b) Busca de custo uniforme;
 - c) Busca em profundidade;
 - d) Busca com aprofundamento iterativo.

Busca em largura (Breadth-First Search - BFS)

- Nó raiz é expandido primeiro, depois todos os seus sucessores, depois os sucessores deles e assim por diante;
- Na busca em largura, todos os nós em um nível da árvore de busca são expandidos antes dos nós do nível seguinte;
- Exemplo: visuAlgo <http://visualgo.net/dfsdfs>
- A fronteira na busca em largura pode ser vista como uma **fila (FIFO)**.

Busca em profundidade (Depth)

- A busca em profundidade sempre expande o nó até mais profundo até chegar no objetivo ou em um **nó folha** (nó sem filhos ou sucessores);
- Se chegar a um nó folha que não é o nó objetivo, a busca em profundidade faz backtracking (retrocesso) para expandir os nós que ainda estão na fronteira do espaço de estados;
- Exemplo: visuAlgo <http://visualgo.net/dfsdfs>
- A fronteira na busca em profundidade pode ser vista como uma pilha (**LIFO**);
- Exemplo: Labirinto
https://www.youtube.com/watch?v=O7pMHx09Z_w

Busca em largura x profundidade

- A busca em largura é completa quando o fator de ramificação é finito;
- A busca em largura é ótima se o custo de caminho cresce com a profundidade do nó;
- Busca em profundidade geralmente é melhor do que a largura;
- A busca em profundidade pode ficar paralisada ao descer em um caminho muito longo (ou infinito), por isso não é completa;
- A busca em profundidade deve ser evitada quando as árvores geradas são muito profundas.

Busca em profundidade limitada

- É a busca em profundidade limitada a um limite máximo de profundidade a ser explorado;
- Com esse limite, resolve-se o problema dos caminhos infinitos;
- A busca em profundidade limitada pode ser incompleta se a solução mais rasa estiver abaixo do limite definido.

Busca por aprofundamento iterativo

- Encontra o melhor limite de profundidade aumentando gradualmente o limite, até encontrar o objetivo;
- É o método de busca sem informação preferido quando existe um espaço de busca grande e a profundidade da solução não é conhecida.

T2

- Trabalho individual;
- Escolher um estado brasileiro;
- Definir 10 cidades deste estado;
- Montar as localizações destas cidades em um grafo;
- Definir busca em largura, executar algoritmo mostrando os estágios partindo de uma cidade até outra (escolher cidades distantes);
- Definir busca em profundidade, executar algoritmo mostrando os estágios partindo de uma cidade até outra (escolher cidades distantes).

Busca com informação

Busca com informação (heurística)

- Expande os nós com base em uma **função de avaliação** $f(n)$;
- Essa função de avaliação mede a distância até o objetivo (considerando a heurística);
- O nó com avaliação **mais baixa** é selecionado para expansão;
- Existem vários algoritmos heurísticos: várias funções de avaliação;
- Para controlar os nós que irão ser expandidos, pode-se utilizar uma **fila de prioridades**.

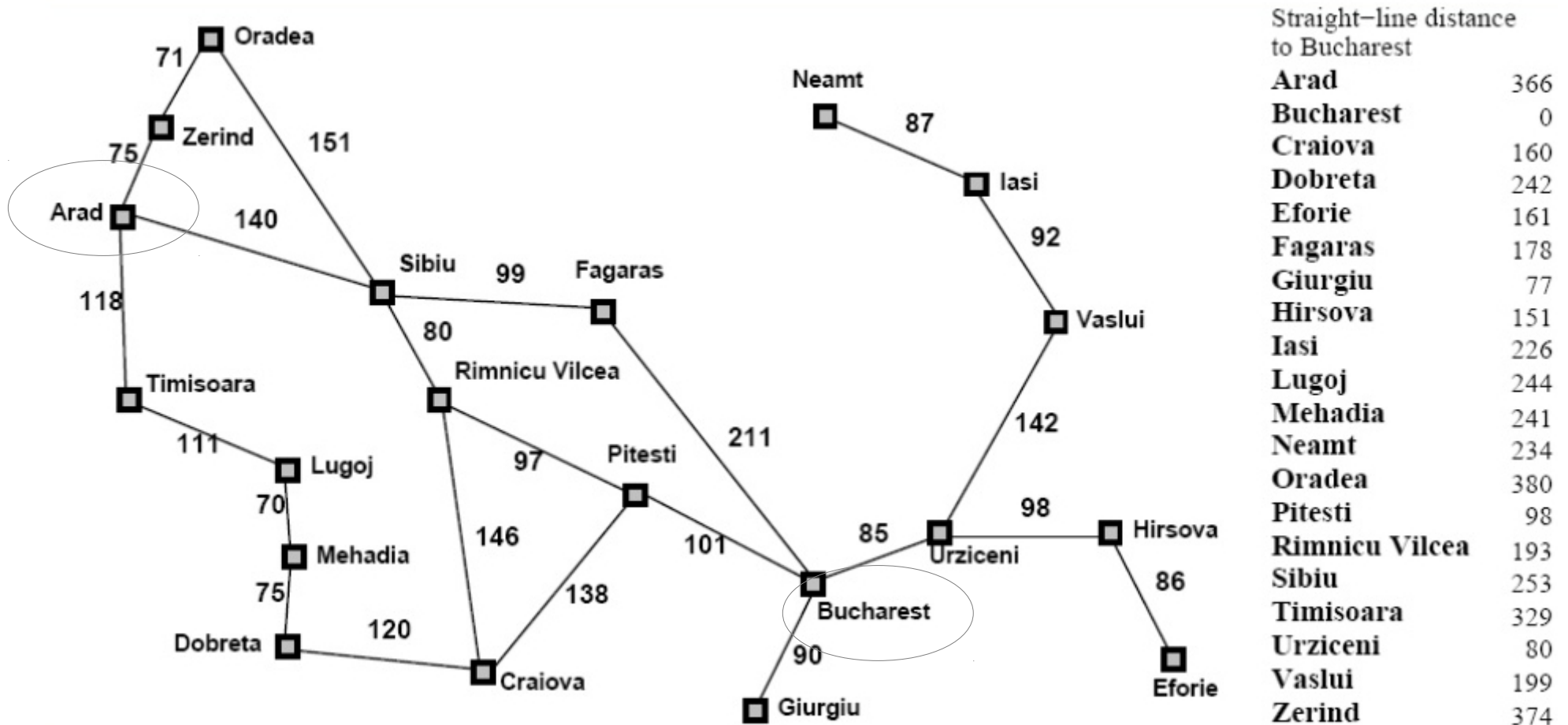
Busca com informação (heurística)

- Pode-se introduzir os nós na fila de prioridades de acordo com o valor da função de avaliação $f(n)$;
- Outro conceito importante é a função heurística $h(n)$, que estima o custo do caminho de menor custo de n até um objetivo.

Busca gulosa (gananciosa ou greedy)

- Tenta expandir o nó mais próximo (“apetitoso”) ao estado atual supondo que provavelmente levará a uma solução rápida;
- Avalia os nós usando apenas a função heurística:
 $f(n)=h(n)$;
- A implementação do algoritmo é similar ao utilizado na busca sem informação, entretanto utiliza-se uma função heurística para decidir qual o nó deve ser expandido;
- Por exemplo, a heurística poderia ser a distância em linha reta de uma cidade X até uma cidade objetivo.

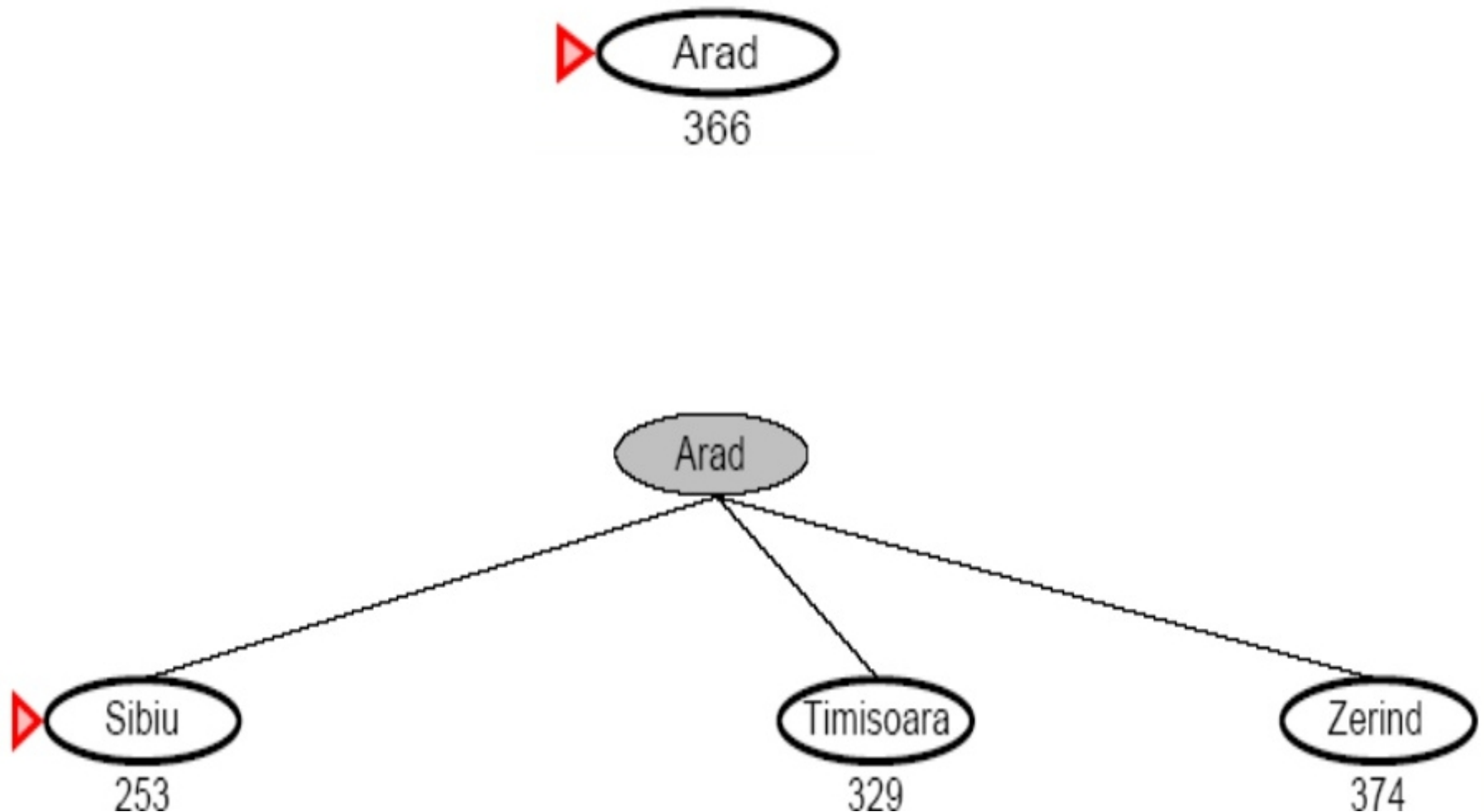
Busca gulosa (gananciosa ou greedy)



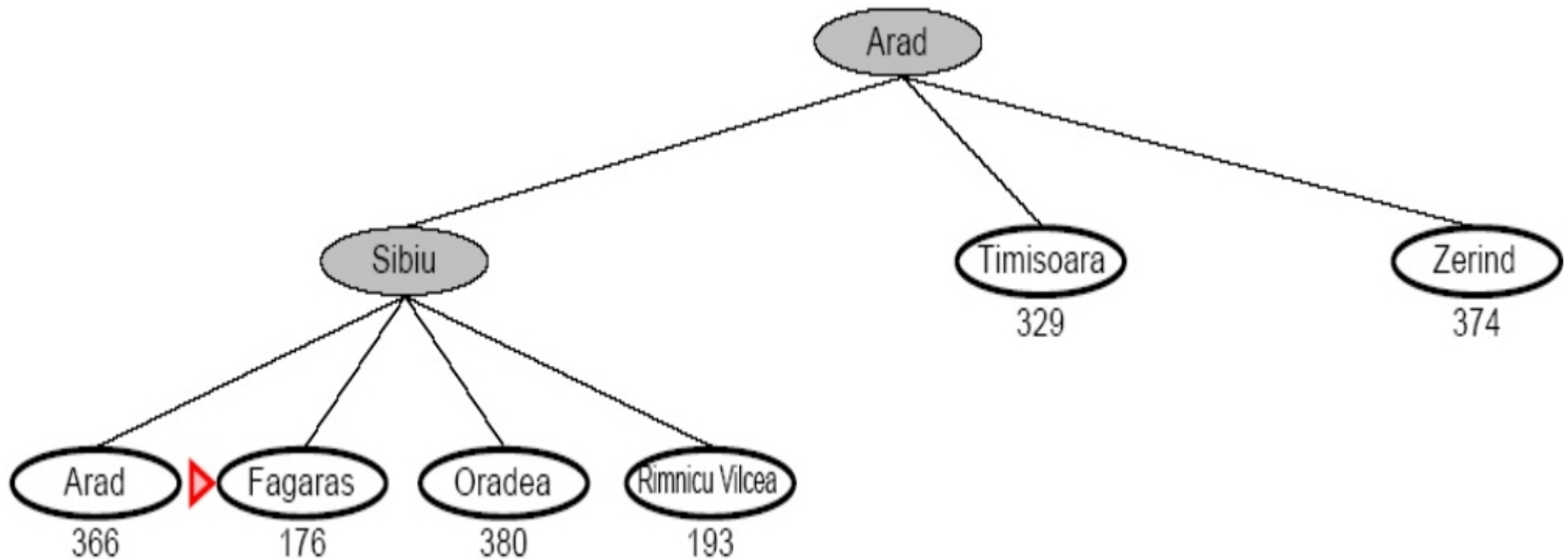
* o cálculo guloso é feito sobre linhas retas até o objetivo.

** os valores no mapa são maiores, pois apesar de serem demonstradas linhas retas, são distâncias reais.

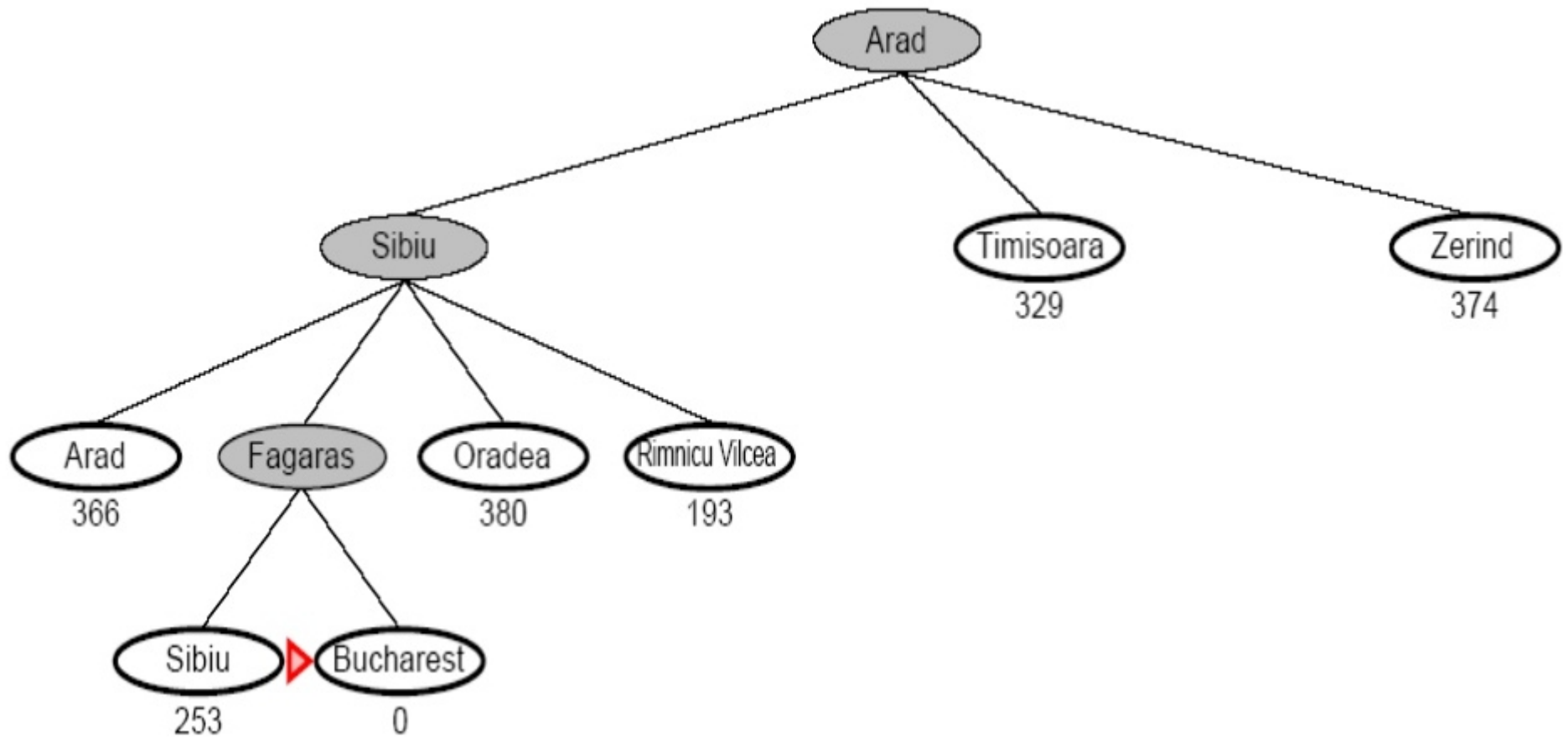
Busca gulosa (gananciosa ou greedy)



Busca gulosa (gananciosa ou greedy)



Busca gulosa (gananciosa ou greedy)



Busca gulosa (gananciosa ou greedy)

O algoritmo guloso **não garante a solução ótima** pois ele escolhe sempre a melhor solução no **momento atual (local)**, mas isso nem sempre é uma solução ótima global;

A busca gulosa é **incompleta**, pois ela pode entrar em um caminho infinito.

Busca A*

- A busca A* avalia os nós combinando:
g(n): custo real do caminho para alcançar cada nó (custo do nó inicial até o nó n);
h(n): custo estimado para ir do nó n até o objetivo.

Busca A*

- $f(n)=g(n)+h(n)$;
- Ideias:
 - evitar expandir caminhos que já ficaram caros;
 - definir primeiro o nó com menor valor $f(n)$

Busca A^*

- Se a função heurística $h(n)$ satisfaz algumas condições, então A^* é completa e ótima;
- É ótima se $h(n)$ for heurística admissível:

Para ser heurística admissível, nunca superestima o custo para alcançar o objetivo, e supõe que o custo da resolução do problema é menor do que ele é na realidade.

- Assim, $f(n)$ nunca irá superestimar o custo verdadeiro de uma solução já que $g(n)$ é o valor exato.

Busca A*

- Heurística consistente: assegura que o caminho ótimo para qualquer estado é sempre o primeiro a ser seguido;
- Se h é consistente, então sempre que o A* expande um nó, ele achou o caminho ótimo até esse nó;
- Uma heurística consistente fica mais precisa a medida que se aprofunda na árvore de busca;
- Toda heurística consistente também é admissível;
- Se $h(n)$ é consistente, então os valores de $f(n)$ ao longo de qualquer caminho são não-decrescentes;
- https://www.youtube.com/watch?v=qiZVRTDb_Ns

Busca A*: exercício 1

- Aplicar o algoritmo A* para achar o caminho de A até I no labirinto:

Regra: se houver empate, optar pela ordem SUL, LESTE, NORTE e OESTE.



Busca A*: exercício 1

- Para facilitar, passar os vértices do problema para numerais:

1	2	3
4	5	6
7	8	9

Busca A*: exercício 1

- A função de avaliação é $f(n)=g(n)+h(n)$, onde:
 - $g(n)$ é o custo do movimento de um nó para outro nó;
 - $g(n)=(g(n) \text{ do pai})+(\text{custo de movimento para um nó})$;
 - O custo de movimento para um nó é 1;
 - $h(n)$ é o valor da heurística;
 - A heurística que será utilizada é a distância de um nó ao nó de destino.

Busca A*: exercício 1

- Valores de $h(n)$ para cada nó:

1	2	3
4	5	6
7	8	9

n	$h(n)$
1	4
2	3
3	4
4	3
5	2
6	1
7	4
8	3
9	0

Busca A*: exercício 1

- Valores de $h(n)$ para cada nó:

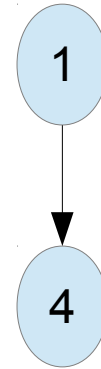
1	2	3
4	5	6
7	8	9

n	$h(n)$
1	4
2	3
3	4
4	3
5	2
6	1
7	4
8	3
9	0

Busca A*

1	2	3
4	5	6
7	8	9

n	h(n)
1	4
2	3
3	4
4	3
5	2
6	1
7	4
8	3
9	0



$$g(1)=0$$

$$g(4)=g(1)+1=0+1=1$$

$$h(4)=3$$

$$f(4)=g(4)+h(4)=1+3=4$$

Busca A*

1	2	3
4	5	6
7	8	9

n	h(n)
1	4
2	3
3	4
4	3
5	2
6	1
7	4
8	3
9	0



$$g(4)=1$$

$$g(5)=g(4)+1=1+1=2$$

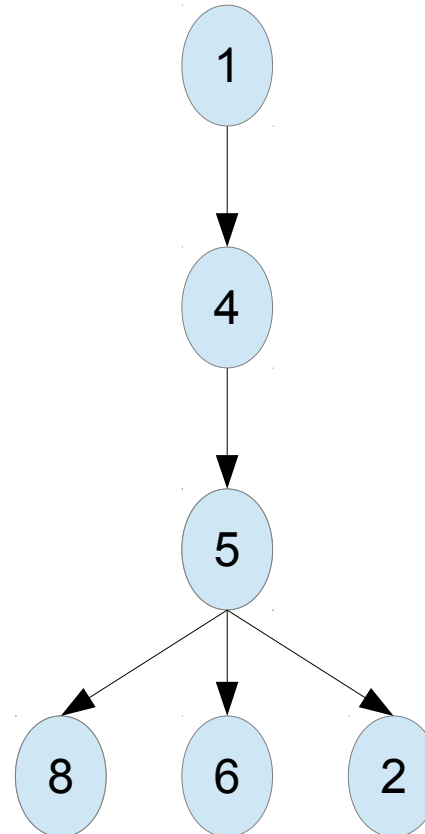
$$h(5)=2$$

$$f(5)=g(5)+h(5)=2+2=4$$

Busca A*: exercício 1

1	2	3
4	5	6
7	8	9

n	h(n)
1	4
2	3
3	4
4	3
5	2
6	1
7	4
8	3
9	0



$$g(5)=2$$

$$f(8)=(g(5)+1)+3=6$$

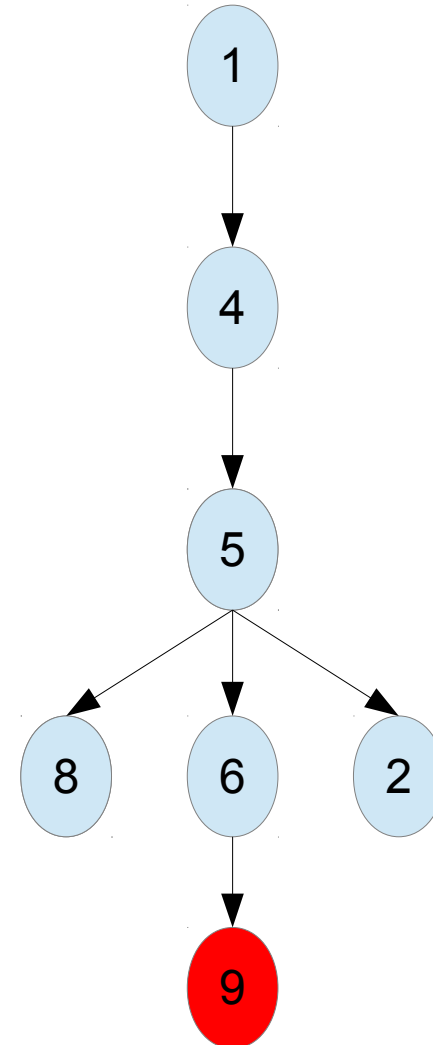
$$f(6)=(g(5)+1)+1=\mathbf{4}$$

$$f(2)=(g(5)+1)+3=6$$

Busca A*: exercício 1

1	2	3
4	5	6
7	8	9

n	h(n)
1	4
2	3
3	4
4	3
5	2
6	1
7	4
8	3
9	0



T3

- Entregar no fim da aula;
- Individual;
- Executar o algoritmo de busca cega em profundidade e em largura para o labirinto anterior.

Outras estratégias de busca

O que foi visto até agora foram buscas sistemáticas, onde é mantido um ou mais caminhos em memória, registrando alternativas exploradas e não exploradas (problema de processamento e memória);

Existem problemas em que o caminho é irrelevante e o estado final em si é a solução (exemplo: problema das 8 rainhas);

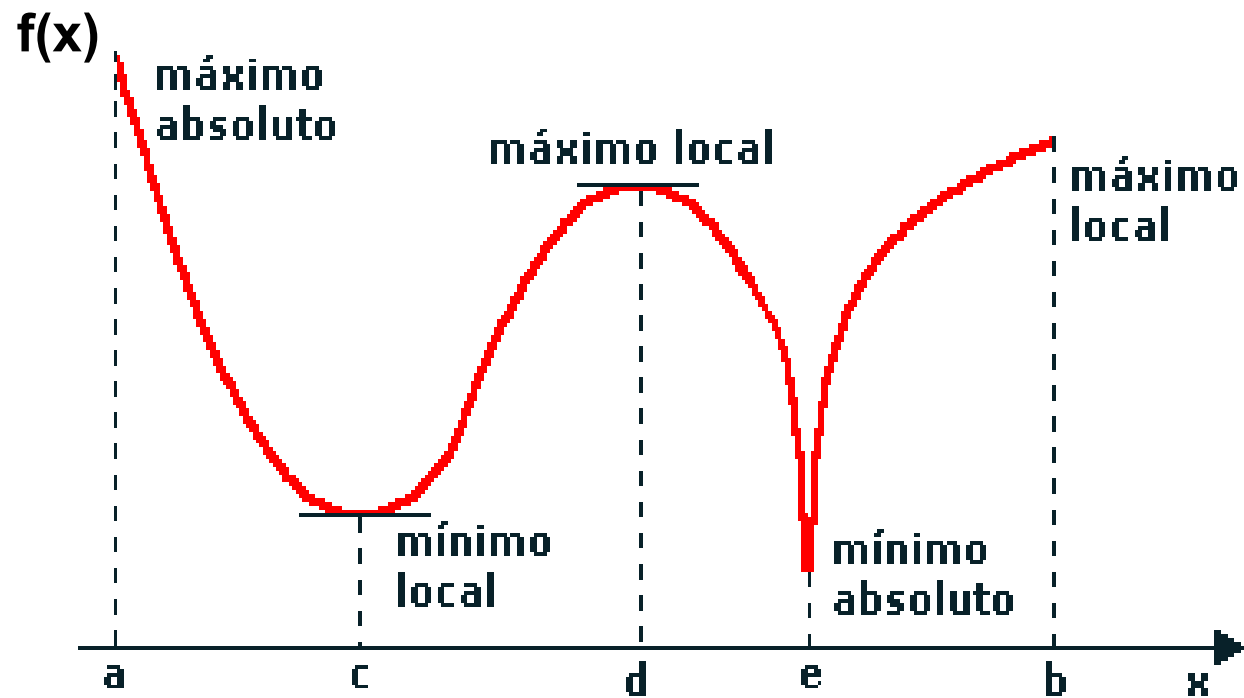
Meta-heurística

- Servem para resolver de forma genérica problemas de busca e otimização;
- Não necessariamente armazenam o caminho à solução!

Otimização

- É um problema qualquer onde deseja-se encontrar um estado que **maximize** ou **minimize** a função objetivo (função de avaliação);
- função objetivo (função de avaliação): diz o quão bom é um determinado estado;
- Um **algoritmo completo** sempre encontra uma solução se ela existir;
- Um **algoritmo ótimo** sempre encontra o mínim/máximo global.

Otimização



Busca com melhoria iterativa

- Exemplos de algoritmos de melhoria iterativa:
 - 1) Hill-Climbing (subida de encosta):
 - Só faz modificações que melhoram o estado atual;
 - 2) Simulated Annealing (têmpera simulada);
 - Pode fazer modificações que pioram temporariamente o estado atual, para possivelmente melhorá-lo no futuro.

Busca Hill-climbing

- O algoritmo se “move” de forma contínua em valor crescente;
- Termina quando alcança um pico em que nenhum vizinho tem valor mais alto;
- Examina vizinhos imediatos;
- Não precisa manter a árvore de busca (menos memória);
- Armazena só o estado corrente e tenta melhorá-lo sucessivamente.

Busca Hill-climbing

Nó_atual=estado_inicial(problema)

Repita

 Nó_vizinho=sucessor de Nó_atual com o melhor valor de avaliação

 Se $\text{aval}(\text{nó_vizinho}) \leq \text{aval}(\text{nó_atual})$ então

 Retorne como solução o estado do nó corrente

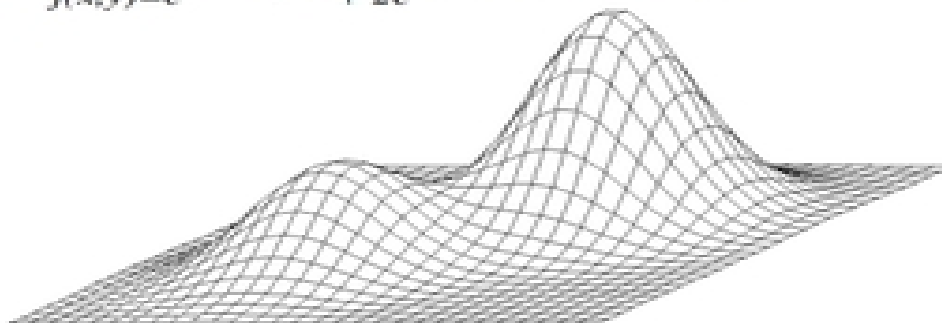
 Nó_atual=Nó_vizinho

Fim_repita

Busca Hill-climbing

- A subida em encosta é também chamada de busca gulosa local;
- Captura um bom estado vizinho, porém pode ficar paralisada em máximos/mínimos locais:

$$f(x,y)=e^{-(x^2+y^2)} + 2e^{-((x-1.7)^2+(y-1.7)^2)}$$

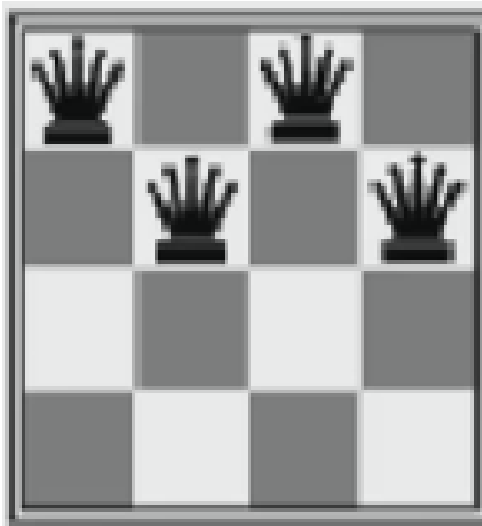


Busca Hill-climbing

- O sucesso da busca em encosta depende da topologia do espaço de estados;
- Se houver poucos máximos locais e platôs, então pode encontrar um máximo local razoavelmente bom;
- <https://www.youtube.com/watch?v=12s-Ilaz0b0>
- <https://www.youtube.com/watch?v=iCMspkozHAI>

Busca Hill-climbing

- Aplicar a busca em hill-climbing no tabuleiro:



x		x	
	x		x

- * Deve-se minimizar o número de pares de rainhas que se atacam;
- ** Função heurística $h = 5$ (número de pares de rainhas que se atacam).

Busca Hill-climbing

x		x	
	x		x

h=5

* escolha dos vizinhos: aleatoriamente, move-se uma rainha 1 casa em uma determinada coluna.

		x	
x	x		x

h=5

x	x	x	
			x

h=4

x		x	
			x
	x		

h=2

x			
	x	x	x

h=4

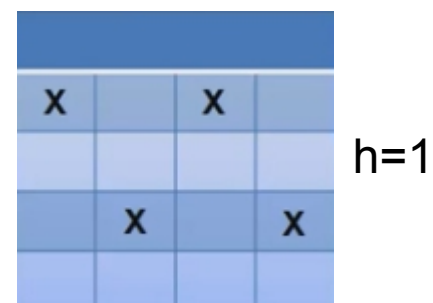
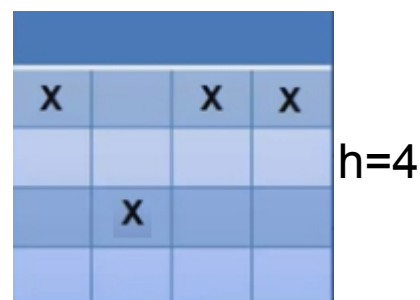
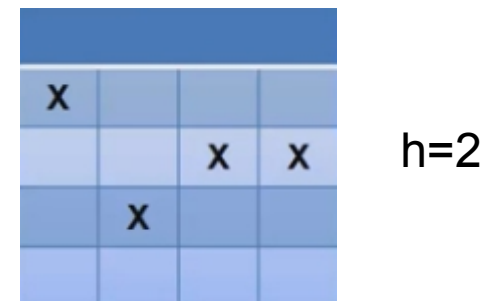
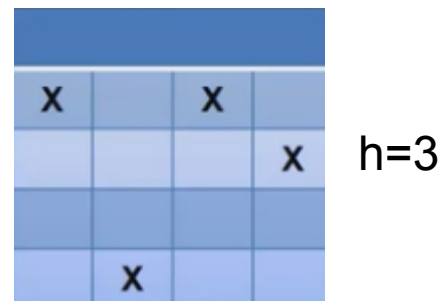
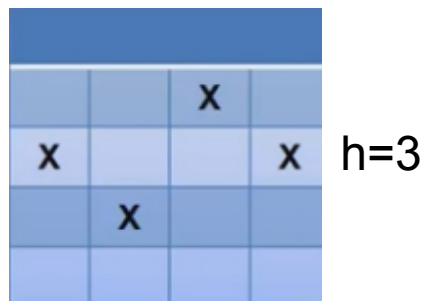
x		x	x
	x		

h=5

x		x	
	x		
			x

h=2

Busca Hill-climbing



* haverá mais uma rodada (com $h=1$), mas nenhum vizinho será melhor, acabando o processo.

Algoritmos genéticos

- Engloba métodos e técnicas computacionais inspiradas na teoria da evolução das espécies (seleção natural) de Darwin;
- A computação se inspira constantemente na biologia.

Algoritmos genéticos

- A 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 (hereditariedade);
- As espécies evoluem pelo princípio da seleção natural e sobrevivência do mais apto;
- Os AGs também são inspirados na genética (Mendel).

Algoritmos genéticos

- John Henry Holland e seus alunos;
- Teoria da evolução das espécies: sobrevivência dos mais aptos;
- Genética: hereditariedade e variabilidade genética para troca de informações visando uma melhoria;



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;
- Um indivíduo no AG é chamado de cromossomo;
- Um indivíduo é um conjunto de atributos da solução, geralmente é uma cadeia de bits que representa uma possível solução para o problema;
- A representação de cada indivíduo depende do problema.

Algoritmos genéticos

- Cada elemento do vetor é um gene, ou seja, um atributo da solução;
- A função de avaliação (função de fitness) determina uma nota a cada indivíduo;
- Essa nota avalia quão bom é aquele indivíduo.

Algoritmos genéticos

- Poderíamos ter como objetivo maximizar o número de 1s:

Indivíduos	Função de aptidão (<i>fitness</i>)
[11101]	4
[01101]	3
[00110]	2
[10011]	3
Aptidão média	3

Algoritmos genéticos

- Segundo Darwin, o melhor sobrevivente é selecionado;
- Existem vários métodos para selecionar o melhor sobrevivente:
 - Exemplos: seleção por roleta ou seleção por torneio.
- A seleção dirige o AG para as melhores regiões do espaço de busca.

Algoritmos genéticos

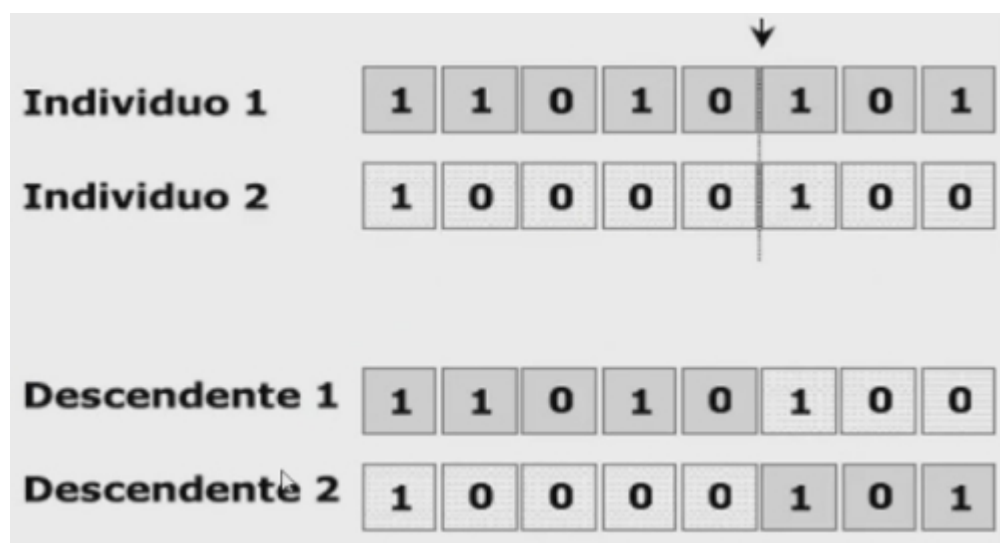
- Método da roleta: cada indivíduo tem uma probabilidade de ser selecionado que é proporcional a sua aptidão;
 - Problema: espaços na roleta podem ser desleais.
- Através da roleta, selecionamos os indivíduos que irão participar do processo de reprodução;

Algoritmos genéticos

- Operadores genéticos (permitem explorar áreas de busca desconhecidas).
 - Cruzamento (crossover): cria novos indivíduos misturando características de dois indivíduos (os pais);
 - O resultado desta operação é um indivíduo que potencialmente combine as melhores características dos indivíduos usados como base.

Algoritmos genéticos

- Cruzamento em um ponto:



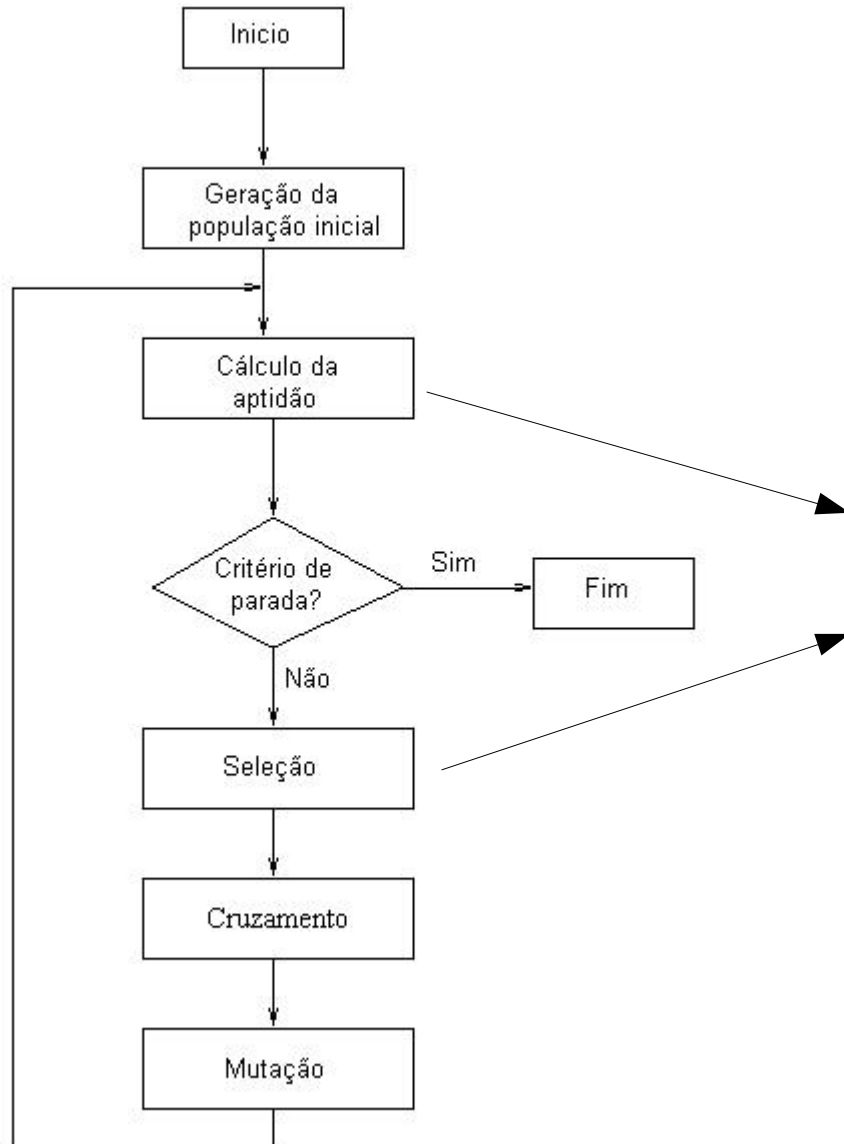
Algoritmos genéticos

- Mutação: modifica aleatoriamente alguma característica do indivíduo;
- O operador de mutação é necessário para a introdução e manutenção da diversidade genética da população;
- A mutação assegura que a probabilidade de se chegar a qualquer ponto do espaço de busca não será zero.

Algoritmos genéticos

- O algoritmo é iterado até um critério de parada;
- A cada passo, um novo conjunto de indivíduos é gerado a partir da população anterior (geração);
- https://www.youtube.com/watch?v=Gl3EjiVlz_4
- <https://www.youtube.com/watch?v=u2t77mQmJiY>
- Código AG básico em java;

Algoritmos genéticos



Utiliza uma função de avaliação para medir a *aptidão* de cada indivíduo.

Algoritmos genéticos

- Exemplo:

população inicial (geração 0):

00100101 (mãe)

10101011 (pai)

geração 1 (após *crossover* direto, *pontuado no bit 5* – até o bit 5 filho herda da mãe, após do pai):

00100011 (filho 1)

00101011 (filho 2)

00100011 (filho 3)...

* no exemplo Java, o crossover acontece por bit, obedecendo o parâmetro `GA.uniformRate` (se `random <= uniformRate`: **mãe**, senão: **pai**)

Algoritmos genéticos

- Seleção direta pai e mãe:

```
pop.saveIndividual(0, pop.getIndividual(0));  
Individual indiv1 = pop.getIndividual(0);  
pop.saveIndividual(1, pop.getIndividual(1));  
Individual indiv2 = pop.getIndividual(1);
```



Selecionando
direto os pais para
o cruzamento...

* O que acontece se tiver mais pais e mães (população tem mais do que 2 elementos por geração) ?

Algoritmos genéticos

- Seleção torneio:

```
int randomId;
for (int k = 0; k < GA.selectionSize; k++) {
    randomId = (int) (Math.random() * pop.size());
    selection.saveIndividual(k, pop.getIndividual(randomId));
}
// get the two best (parents) sorting selection
Individual aux;
for (int g = 0; g < selection.size(); g++) {
    for (int j = 0; j < selection.size(); j++) {
        if (selection.individuals[g].getFitness() >=
selection.individuals[j].getFitness()) {
            aux = selection.individuals[g];
            selection.individuals[g] = selection.individuals[j];
            selection.individuals[j] = aux;
        }
    }
}

Individual indiv1 = selection.individuals[0];
selection.saveIndividual(0, indiv1);
Individual indiv2 = selection.individuals[1];
selection.saveIndividual(1, indiv2);
```

Algoritmos genéticos

- Mutação

```
for (int i = 0; i < indiv.size(); i++) {  
    if (Math.random() <= GA.mutationRate) {  
        // Create random gene  
        byte gene = (byte) Math.round(Math.random());  
        indiv.setGene(i, gene);  
    }  
}
```

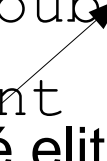
* O que acontece se não tiver mutação através das gerações ?

Algoritmos genéticos

- Altere os parâmetros e explique o que acontece

```
public static final int generationSize = 5;  
public static final int populationSize = 10;  
public static final String solution =  
    "000000000000";  
public static final double uniformRate = 0.5;  
public static final double mutationRate = 0.015;  
public static final int selectionSize = 4;  
public static final boolean elitism = false;
```

O que é elitismo?



Simulated Annealing

- **Otimização** é a escolha do melhor elemento em um conjunto de alternativas disponíveis.
O melhor elemento pode variar de um simples valor que maximiza ou minimiza uma função até estruturas mais complexas (exemplo: melhor arranjo de um conjunto de variáveis).
- A área da otimização é composta por várias subáreas:
 - a) Programação inteira (valores inteiros);
 - b) Otimização estocástica (variáveis aleatórias).

Simulated Annealing

- **Algoritmos exatos:**

Vantagem: garante a solução ótima.

Desvantagens:

Custo de tempo;

Modelagem complexa.

- **Heurísticas:**

Heurísticas são técnicas que buscam boas soluções a um custo computacional razoável de fácil implementação;

As heurísticas não garantem a **otimalidade**;

A desvantagem das heurísticas é a dificuldade de se escapar de ótimos locais;

A metodologia metaheurística surgiu para possibilitar sair desses ótimos locais permitindo a busca em regiões mais promissoras

Simulated Annealing

- **Metaheurísticas** são procedimentos que empregam estratégias para escapar de mínimos locais em espaços de busca de soluções complexas.

Uma metaheurística visa a produzir um resultado satisfatório para um problema, mas não garante a otimalidade;

Metaheurísticas são aplicadas para encontrar respostas a problemas sobre os quais há poucas informações e que a estratégia de força bruta é desconsiderada por conta do espaço de solução ser muito grande;

Exemplos de metaheurísticas: Tabu Search, Simulated Annealing, Algoritmos Genéticos, Ant Colony Optimization.

Simulated Annealing

- Vizinhaça: um vizinho de uma solução S é uma solução S' na qual foi aplicado um movimento (definido anteriormente) modificando a solução corrente.

Simulated Annealing

- Trata-se de um algoritmo de busca local baseado no conceito de recozimento (“annealing”).
- O processo de recozimento consiste em aquecer um metal até o ponto de fusão e então resfriá-lo lentamente permitindo que suas moléculas alcancem uma configuração de baixa energia e formem uma estrutura cristalina, livre de defeitos.
- **Simulated Annealing (“recozimento simulado”)** estabelece uma conexão entre o comportamento termodinâmico e a busca pelo máximo/mínimo global de um problema de otimização discreto.

Simulated Annealing

- A cada iteração, a função objetivo gera valores para duas soluções:

A atual e a escolhida. Essas soluções são comparadas e, então, as soluções melhores que a atual são sempre aceitas, enquanto que uma fração das soluções piores que a atual são aceitas na esperança de se escapar de um mínimo/máximo local.

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.

- <https://www.youtube.com/watch?v=rsGOB80v0-k>
- <https://www.youtube.com/watch?v=SC5CX8drAtU>

T5

- Entender o que é Simulated Annealing, para explicar o funcionamento do código TSP passado em aula.

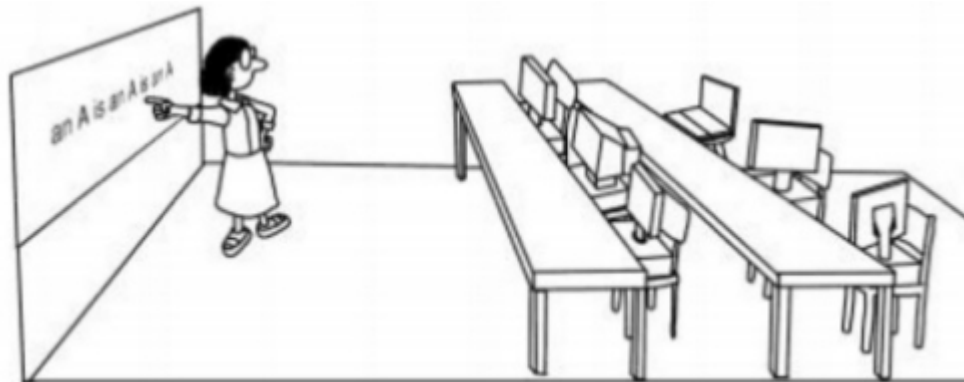
Aprendizagem de máquina

Aprendizagem de máquina

- Formas de aprendizado:
 - **Aprendizado supervisionado:** existe um “professor” que avalia a resposta da saída. Ex: árvores de decisão, KNN, SVM, redes neurais, entre outros.
 - **Aprendizado não supervisionado:** não existe “professor”, a própria técnica deve avaliar suas relações, padrões, irregularidades, etc, de forma a obter sua saída. Ex: PCA, Clustering, Análise competitiva, K-Means, ART, Self-Organizing Maps, entre outros.
 - Aprendizado por reforço.

Aprendizagem de máquina

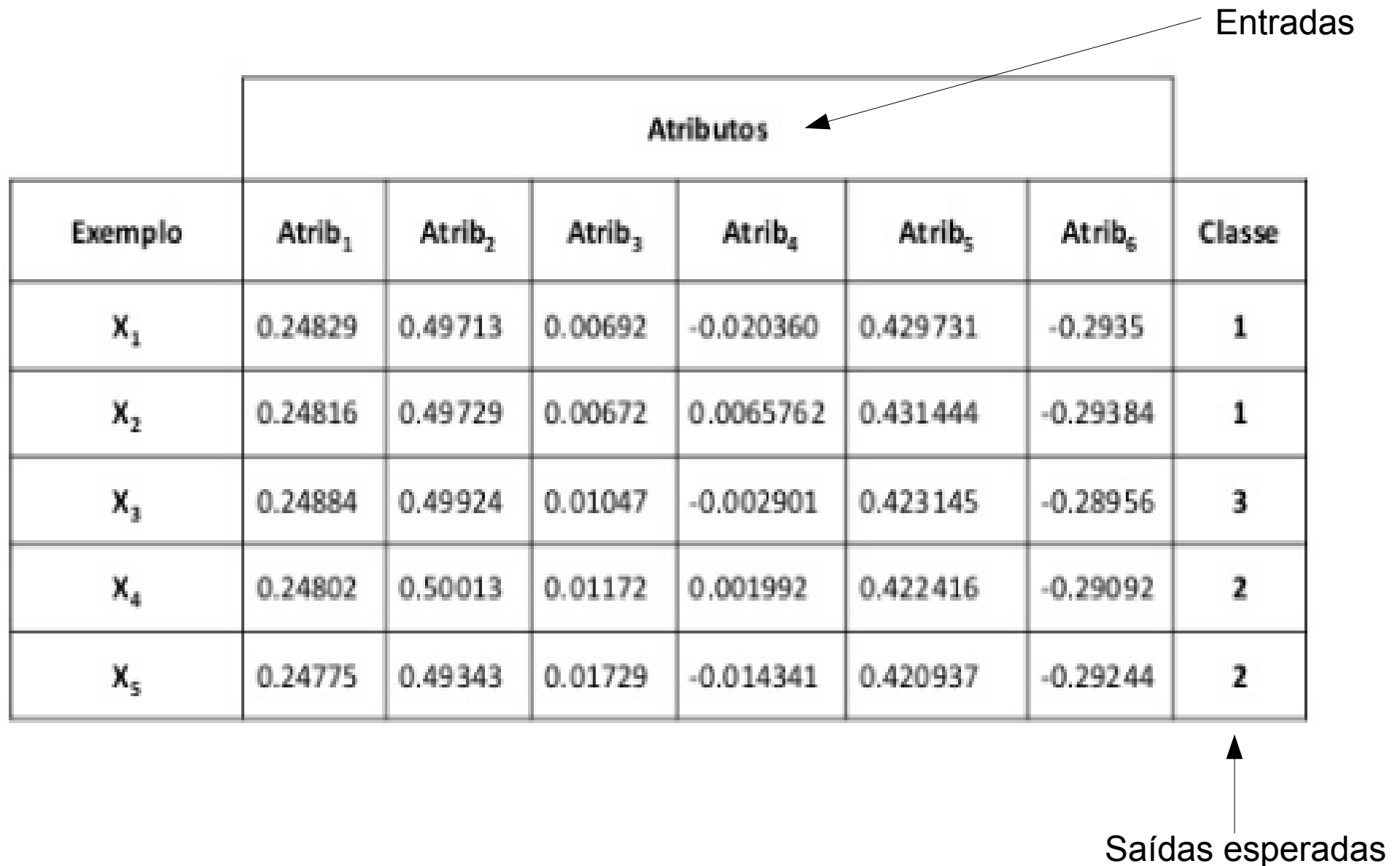
- **Aprendizado supervisionado:** apresentam-se as entradas e as saídas ao problema. Ex. Classificação.



Aprendizagem de máquina

- Aprendizagem não é “decorar”. Qualquer computador consegue memorizar informação (muito melhor do que o humano). Máquinas precisam “**generalizar**” a informação.
- A **generalização** é a capacidade de uma máquina exibir um comportamento esperado de acordo com uma nova situação.

Aprendizagem de máquina



Atributos							
Exemplo	Atrib ₁	Atrib ₂	Atrib ₃	Atrib ₄	Atrib ₅	Atrib ₆	Classe
X ₁	0.24829	0.49713	0.00692	-0.020360	0.429731	-0.2935	1
X ₂	0.24816	0.49729	0.00672	0.0065762	0.431444	-0.29384	1
X ₃	0.24884	0.49924	0.01047	-0.002901	0.423145	-0.28956	3
X ₄	0.24802	0.50013	0.01172	0.001992	0.422416	-0.29092	2
X ₅	0.24775	0.49343	0.01729	-0.014341	0.420937	-0.29244	2

Aprendizagem de máquina

- **Modelo**: ao “treinar” uma técnica com “entradas” e “saídas esperadas”, é construído um modelo (“**armazenador de conhecimento**”). Este modelo será utilizado para as futuras entradas e saídas;

Normalmente, diz-se que o modelo deve ser “treinado” (ou fazer com que ele “aprenda”);

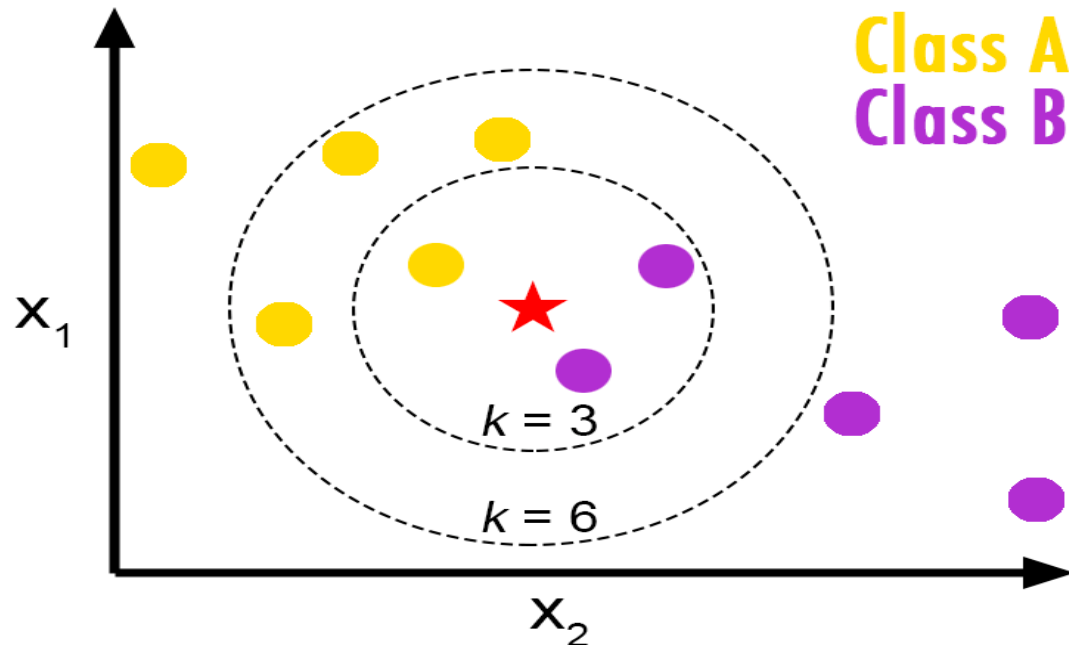
- **Overfit** (sobre-ajuste): quando um modelo se ajuda em demasiado. Baixo poder de generalização;
- **Underfit** (sub-ajuste): o modelo ajusta-se muito pouco em relação aos dados de entrada e saída esperada.

T6

- Fazer um classificador com 50 exemplos, contendo:
 - 5 classes;
 - 5 atributos;
- Pesquisar o que é uma base de treinamento, validação e de testes;
- Classificar exemplos que existam no treinamento, e “tentar” classificar exemplos que não existam no treinamento;

KNN

- KNN: K-nearest-neighbor (K-vizinho-próximo);
- Utilizado para classificar objetos com base em exemplos de treinamento que estão mais próximos no espaço de características;
- O parâmetro k é o número de vizinhos mais próximos.



KNN

- Para utilizar o KNN é necessário:
 - Um conjunto de exemplos para treinamento;
 - Definir uma métrica para calcular a distância entre os exemplos de treinamento;
 - Definir o valor de K.

KNN

- Classificar um exemplo desconhecido com o algoritmo KNN consiste em:
 - Calcular a distância entre o objeto desconhecido e os outros exemplos do conjunto de treinamento;
 - Identificar os K vizinhos mais próximos;
 - Utilizar o “rótulo” (nome) da classe dos vizinhos mais próximos para determinar o rótulo de classe do exemplo desconhecido (votação **majoritária**).

KNN

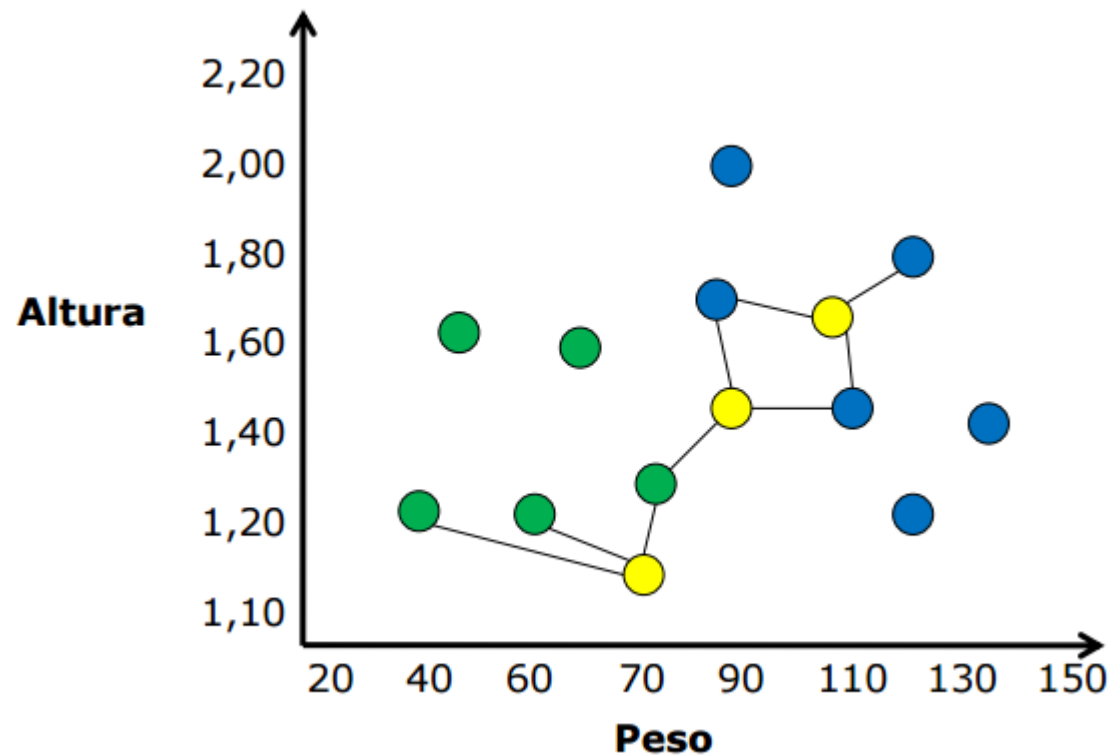
- Calculo da distância entre dois pontos:
 - A mais simples é a distancia euclidiana:

$$d(p, q) = \sqrt{\sum_{i=1}^n (p_i - q_i)^2}$$

- Outras formas de medir distâncias:
 - Distância de Mahalanobis, Minkowsky, Hamming, entre outros.

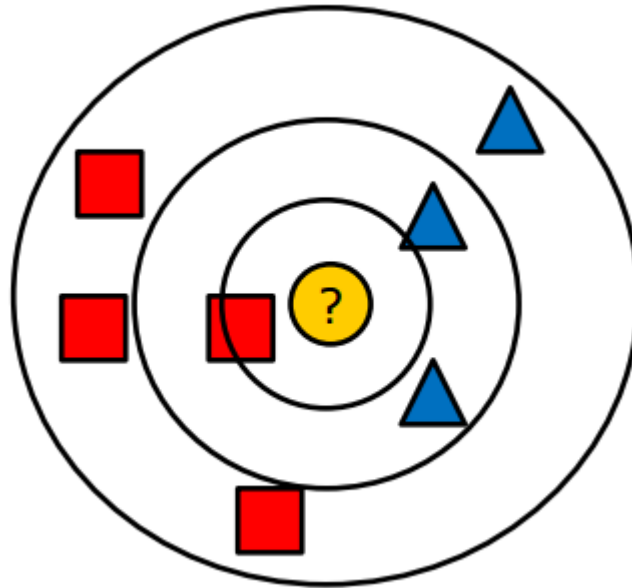
KNN

- Exercício: classificar os elementos em amarelo:



KNN

- Como escolher o valor de K?



KNN

- Como escolher o valor de K?
 - Se K for muito pequeno, a classificação fica sensível a pontos de ruído;
 - Se k for elevado, a vizinhança pode incluir elementos de outras classes.
- * normalmente, escolhe-se um valor ímpar para K (evita empates desnecessários no cálculo).

KNN

- Classificação de base de dados iris dataset.



SETOSA



VERSICOLOR



VIRGINICA

KNN



Features: tamanhos em cms das pétalas e sépalas...

```
class;sepal length; sepal width; petal length; petal width
Iris-setosa;5.1;3.5;1.4;0.2
Iris-setosa;4.9;3.0;1.4;0.2
Iris-setosa;4.7;3.2;1.3;0.2
Iris-setosa;4.6;3.1;1.5;0.2
Iris-setosa;5.0;3.6;1.4;0.2
Iris-setosa;5.4;3.9;1.7;0.4
Iris-setosa;4.6;3.4;1.4;0.3
Iris-setosa;5.0;3.4;1.5;0.2
Iris-setosa;4.4;2.9;1.4;0.2
Iris-setosa;4.9;3.1;1.5;0.1
.....
```

T8

- INDIVIDUAL, entrega (2,5 semanas, contados a partir de hoje);
- Valor: 3 pontos, sendo 1 do relatório + 2 apresentação;
- Fazer um programa em JAVA que utilize a técnica do KNN para classificar padrões;
- Escolher um dataset (de treinamento), utilizar pedaço deste dataset para testes – excluir os padrões de teste do treinamento;
- Escolher um dataset com mais de 2 dimensões;
- Fazer um relatório das atividades e resultados alcançados;
- Não utilizar **nenhuma** biblioteca pronta de inteligência artificial;
- No dia da apresentação, será visto o código e questionamentos.

T7 (com o professor)

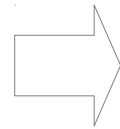
- Tirar 3 fotos “quase 3x4” (centralizar você, e ficar com o rosto “sério”, de frente, com o rosto virado para o lado direito e esquerdo);
- Deixar a foto com tamanho: 430x250 (aproximadamente);
- Utilizar <http://www.faceplusplus.com/demo-detect/> para pegar as features (atributos);
- Montar um exemplo com sua foto, com os seguintes atributos retirados do site (posições de elementos do rosto).



T7 (com o professor)

...

```
"position": {,  
  "eye_left": {  
    "x": 34.843614,  
    "y": 42.830425  
  },  
  "eye_right": {  
    "x": 69.704418,  
    "y": 43.143632  
  },  
  "height":  
42.688679,  
  "mouth_left": {  
    "x": 38.836426,  
    "y": 65.044811  
  },  
  "mouth_right": {  
    "x": 67.186747,  
    "y": 64.666274  
  },  
}
```



```
Eye Left X;Eye Left Y;Eye Right X;Eye  
Right Y;Mouth Left X;Mouth Left  
Y;Mouse Right X;Mouse Right Y;Classe  
34,843614;42,830425;69,704418;43,14363  
2;38,836426;65,044811;67,186747;64,666  
274;André
```

...

T7 (com o professor)

- A sua foto, mais a dos seus colegas formarão a base de treinamento, exemplo:

Eye Left X	Eye Left Y	Eye Right X	Eye Right Y	Mouth Left X	Mouth Left Y	Mouth Right X	Mouth Right Y	Classe
34,843614	42,830425	69,704418	43,143632	38,836426	65,044811	67,186747	64,666274	André
37,702195	28,922195	57,73	29,049756	49,603171	52,39878	55,767561	52,719024	Tom Cruise
40,959	43,578962	58,19667	42,534153	43,3445	72,729781	56,8745	72,358743	Dicaprio
44,326333	21,58262	64,340167	26,901597	44,076333	61,689776	60,564667	64,060064	Bradley Cooper
55,2495	51,273434	71,035333	52,795739	55,1535	80,466165	69,712	79,524561	Ben Stiller

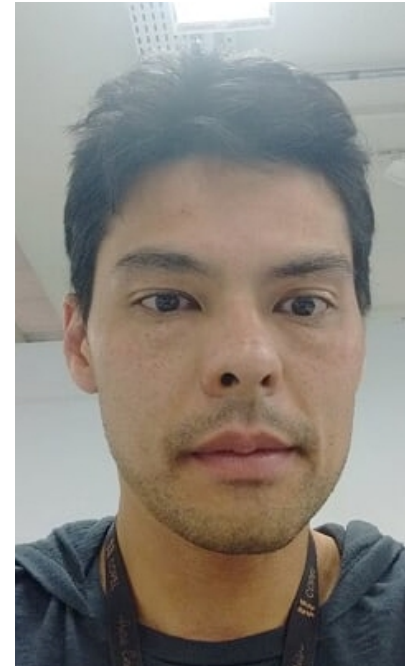
T7 (com o professor)

- Como o tamanho das fotos diferem, vamos tirar a diferença entre as posições das características físicas, por exemplo: distância entre um olho e outro (** considerando que as cabeças nas fotos tem o mesmo tamanho)...

Eyes Distance X	Eyes Distance Y	Mouth Corner Distance X	Mouth Corner Distance Y	Classe
34,860804	0,313207	28,350321	-0,378537	André
20,027805	0,127561	6,16439	0,320244	Tom Cruise
17,23767	-1,044809	13,53	-0,371038	Dicaprio
20,013834	5,318977	16,488334	2,370288	Bradley Cooper
15,785833	1,522305	14,5585	-0,941604	Ben Stiller

T7 (com o professor)

- Vamos tentar classificar a imagem:



- Que possui as características:

Eyes Distance X	Eyes Distance Y	Mouth Corner Distance X	Mouth Corner Distance Y	Classe
35,167699	0,292067	29,742678	-0,153846	?