Inteligência Artificial

Profa. Dra. Alynne Oya Kitamura
Ciência da Computação – UNIP – 5 º semestre
profa.alynne@gmail.com

Conteúdo Programático

Conteúdo Programático

- Introdução: Introdução a Inteligência Artificial; Áreas e Sub-áreas da Inteligência Artificial; Pesquisas realizadas.
- Softwares Inteligentes: Apresentação de Softwares Inteligentes; Comentários sobre os softwares.
- Espaço de Estados: Definição; Representação; Resolução de problemas; Aplicação em exercícios de lógica
- Busca em Largura: Definição; Exemplos; Algoritmo de Busca; Análise da Complexidade; Exercícios

Conteúdo Programático

- Busca em Profundidade e Profundidade Limitada: Definição; Exemplos; Algoritmo de Busca; Análise da Complexidade; Exercícios
- Busca em Profundidade Iterativa: Definição; Exemplos; Algoritmo de Busca; Análise da Complexidade; Exercícios
- Heurísticas e Busca Best-First e Busca A*: Definição; Exemplos; Algoritmo de Busca; Análise da Complexidade; Exercícios
- Busca IDA e Busca Recursiva Best-first (RBFS): Definição; Exemplos; Algoritmo de Busca; Análise da Complexidade; Exercícios

Conteúdo Programático

- Busca Escalada na Montanha (Hill-climbing) e Algoritmos Genéticos: Definição; Exemplos; Algoritmo de Busca; Análise da Complexidade; Exercícios

Bibliografia Básica

- ARTERO. Almir Olivette. Inteligência Artificial Teórica e Prática. São Paulo: Editora Livraria da Física, 2009.

- LUGER, George F. - Inteligência Artificial: Estruturas e Estratégias para a Solução de Problemas Complexos. 4a. Ed. – Ed. Bookman, 2004.

- RUSSELL, Stuart J. NORVIG, Peter. - Inteligência Artificial. – Ed. Campus, 2004.

Bibliografia Complementar

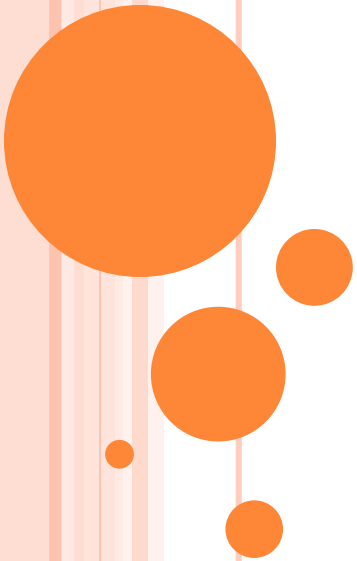
- FERNANDES, Anita Maria Da Rocha - Inteligência Artificial - Noções Gerais – 2003 - Visual Books.

- PESSIS-PASTERNAK, G. Do Caos à Inteligência Artificial. Editora: Unesp, 2001.

- REZENDE S. - Sistemas Inteligentes: Fundamentos e Aplicações. – Ed. Manole, 2002.

- RICH, E. Inteligência Artificial. São Paulo: McGraw-Hill Ltda, 1988.

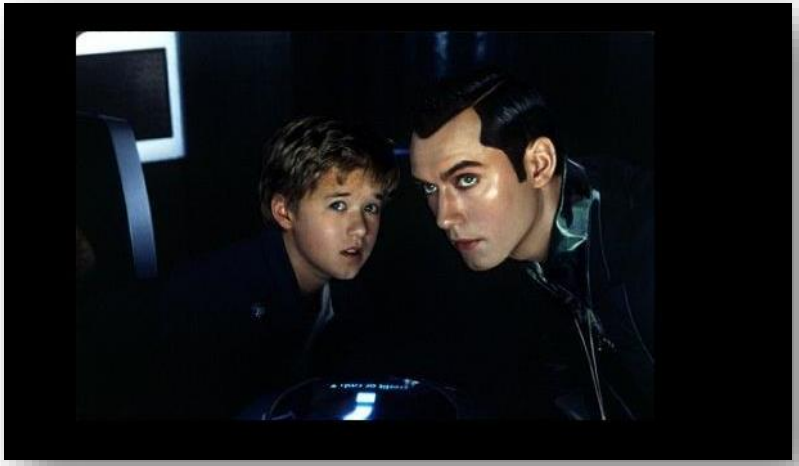
- COPPIN, Ben. Inteligência Artificial. Coleção Illuminated. Ed. LTC, 2010.

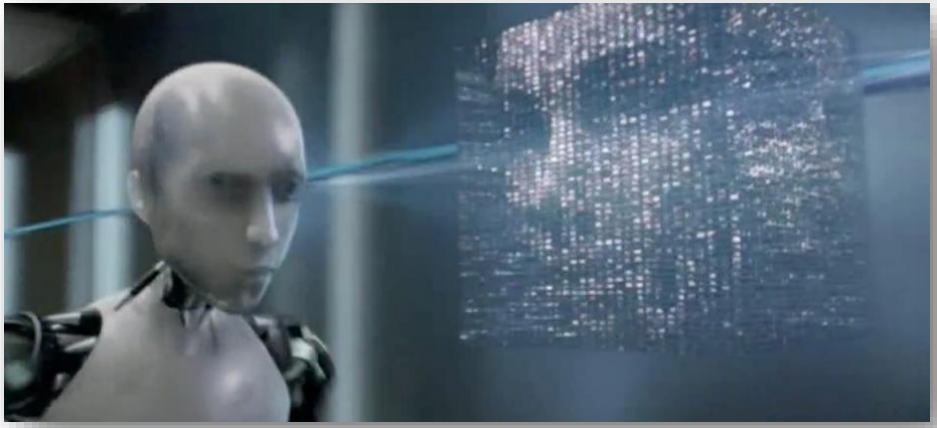


Introdução

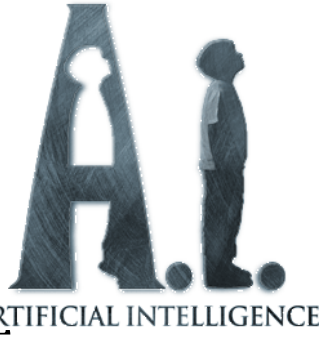
Softwares Inteligens

Inteligência Artificial





Inteligência Artificial

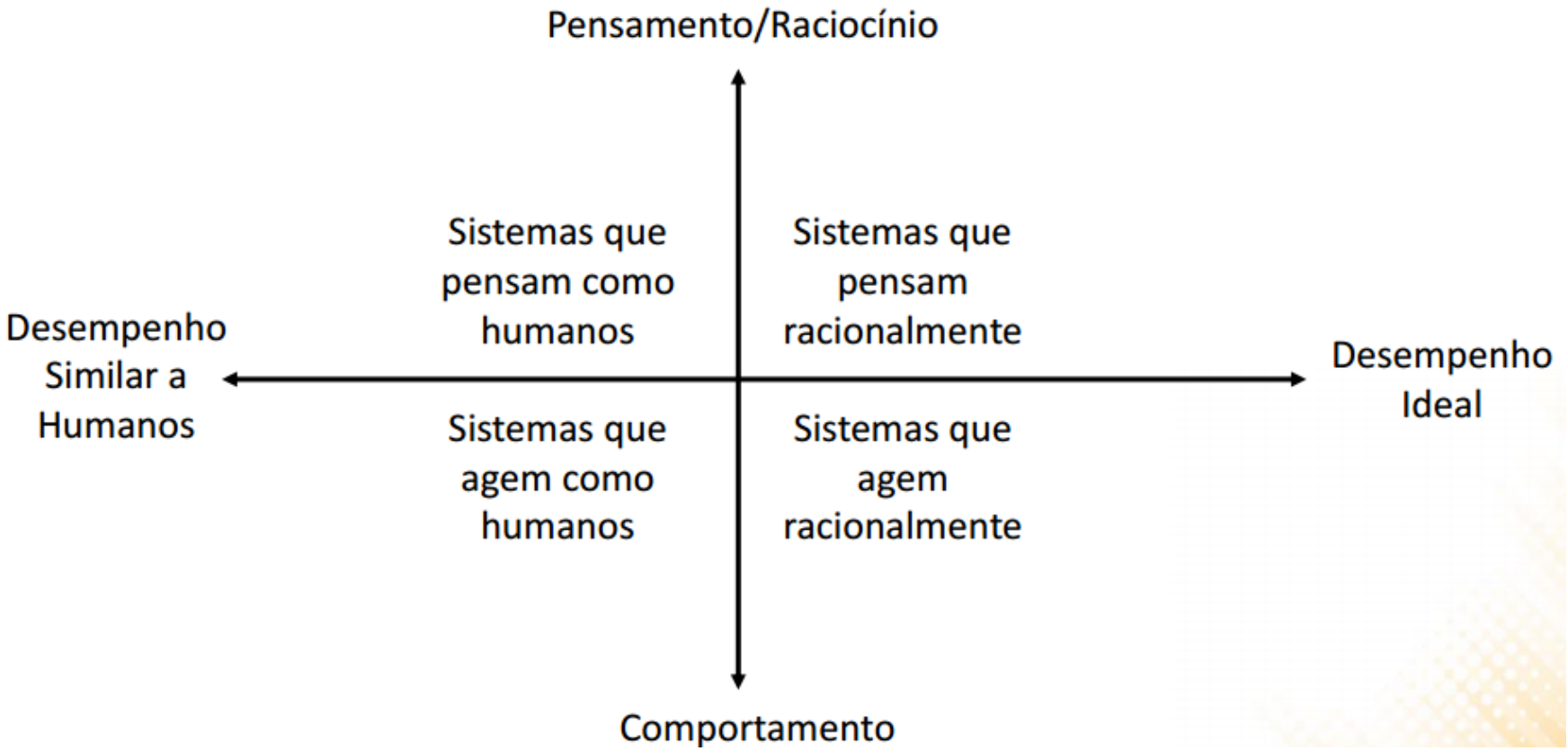


- **Inteligência**: aculdade de entender, pensar, raciocinar e interpretar; entendimento, intelecto. 2 Compreensão, conhecimento profundo. 3 ***Filos*** Princípio espiritual e abstrato considerado como a fonte de toda a intelectualidade. 4 ***Psicol*** Capacidade de resolver situações novas com rapidez e êxito (medido na execução de tarefas que envolvam apreensão de relações abstratas) e, bem assim, de aprender, para que essas situações possam ser bem resolvidas. 5 Pessoa de grande esfera intelectual.
- **Artificial**: Produzido por arte ou indústria do homem e não por causas naturais. **2** Produzido ou efetuado pela habilidade do homem para imitar a natureza.

Inteligência Artificial

- “É o estudo de como fazer os computadores realizarem tarefas as quais, até o momento, os homens fazem melhor” (Rich, 1994).
- "Inteligência artificial é o estudo das ideias que permitem aos computadores serem inteligentes" (Winston, 1984).
- “É o campo de estudo que tenta explicar e simular o comportamento inteligente em termos de processos computacionais” (Schalkoff, 1990).
- “A arte de criar máquinas que executam funções que requerem inteligência quando executadas por pessoas” (Kurzweil, 1990).

Inteligência Artificial



Inteligência Artificial

Russell & Norvig (2003)

- Sistemas que pensam como humanos
  - Ciência Cognitiva
- Sistemas que agem como humanos
  - Teste de Turing
- Sistemas que pensam racionalmente
  - Lógica
- Sistemas que agem racionalmente
  - Agentes racionais – Agem sempre racionalmente para alcançar as suas metas

História da IA

- Inicio dos anos 40 - Segunda Guerra Mundial.
  - Criação dos primeiros computadores.
  - Simulação de guerra.
- 1943 - McCulloch e Pitts realizam os primeiros estudos para criar um modelo de neurônio artificial capaz de resolver qualquer função computável.
- 1956 - Criado oficialmente o termo Inteligência Artificial em um congresso
- no Dartmouth College.
- 1956 - 1966 - Época de sonhos.
  - General Problem Solver (GPS).
  - Lisp - A linguagem Lisp foi projetada primariamente para o processamento de dados simbólicos

História da IA

- 1966 - 1974 - Uma dose de realidade.
  - Livro de Minsky e Papert (1969) critica perceptrons e paralisa investimentos em redes neurais por quase 20 anos (tema volta em 1986).
  - Problema da complexidade computacional do algoritmos.
- 1969 - 1979 - Sistemas baseados em Conhecimento
  - Grande evolução da Inteligência Artificial Simbólica.
  - Desenvolvimento de sistemas especialistas.
  - Prolog. – Linguagem puramente lógica
- 1980 - 1988 - Inteligência Artificial na Indústria
  - Sistema especialistas.
  - Ressurgem as redes neurais.
  - Lógica Fuzzy.

História da IA

- Inicio dos anos 90:
  - Sistemas Especialistas com alto custo de manutenção. Erro foi não ver que o objetivo deve ser Auxiliar, ao invés de Automatizar.
  - Grandes avanços em todas as áreas da inteligência artificial, com manifestações significativas na aprendizagem de máquina, planejamento multi-agente, raciocínio com incerteza, mineração de dados, entre outros tópicos.
- 1997 - Deep Blue (IBM) derrota o campeão mundial de xadrez (Garry Kasparov).
  - Algoritmos de busca, computadores de alta velocidade e hardware desenvolvido especificamente para xadrez.

História da IA

- 2011 – Watson (IBM) derrota os dois maiores jogadores de Jeopardy (Brad Rutter e Ken Jennings).
  - Baseado em tecnicas avançadas de Processamento de Linguagem Natural, Recuperação de Informação, Representação de Conhecimento, Raciocínio e Aprendizado de Máquina.
  - Processamento paralelo massivo.
  - 90 clusters com um total de 2880 servidores com processadores de 3.5 GHz (8 núcleos e 4 threads por núcleo) 16 Terabytes de memória RAM.

Sub- Áreas da Inteligência Artificial

- Redes Neurais Artificiais;
- Computação Evolucionária;
- Lógica Fuzzy;
- Raciocínio Baseado em Casos;
- Sistemas Baseados em Conhecimento;
- Robótica;
- Agentes Inteligentes;
- Processamento de Linguagem Natural;
- Mineração de dados;
- Mineração de textos;
- Outros.

EXAME.COM

Assine

Inteligência artificial do Google é agressiva quando estressada

Em competição, algoritmo da DeepMind adotou estratégia altamente agressiva para vencer

Por Lucas Agrela

16 fev 2017, 14h22 - Atualizado em 16 fev 2017, 17h25

DEEPMIND

Samsung pode gastar US$ 1 bi na compra de empresas de Inteligência Artificial

Por Redação | em 21.02.2017 às 11h36

Para Eric Schmidt, aspectos positivos da inteligência artificial superam os negativos

O CEO da Alphabet, holding do Google, diz que eventuais ameaças de robôs e inteligência artificial são questões filosóficas importantes, mas que nós não estamos enfrentando agora

Erivelto Tadeu - de San
16 de Fevereiro de 2017 -

Recomendadas

E-commerce brasileiro inicia ano com preços em queda


Uber cria comitê

A Microsoft vem investindo pesado no processamento em nuvem Azure e nas máquinas de aprendizado em conjunto com a assistente virtual Cortana. O CEO da companhia, Satya Nadella, está tão satisfeito com os resultados e com os próximos projeto que chamou a inteligência artificial (ou artificial intelligence — AI) de "descoberta suprema", durante um bate-papo sobre tecnologia na segunda-feira (20).

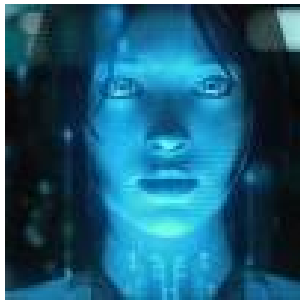
"Estamos dando à (assistente virtual) Cortana uma compreensão de linguagem muito natural"

Nadella atribuiu a AI o papel de maior interface de comunicação já criada até hoje. "Se o sistema operacional era o primeiro tempo de execução (período em que um aplicativo desenvolve sua tarefa), você pode dizer que o segundo era o navegador de web e o terceiro pode ser considerado o agente.

Santander investe em empresas de inteligência artificial, dizem fontes

O braço de capital de risco do banco, Santander InnoVentures, comprou participações na Personetics Technologies

Oito pontos que Marketing e TI precisam saber sobre **Inteligência** ...
CIO - 14 de fev de 2017
E agora surge a **Inteligência Artificial** e o Machine Learning prometendo ajudar os profissionais de marketing a extrair melhores insights dos ...



Ensinar 'bom senso' ou 'ponderação' é desafio da **inteligência artificial**
OK Norton
Folha de S.Paulo - 29 de jan de 2017
A **inteligência artificial** já está entre nós. Muita gente ainda não se dá conta, mas aos poucos ela já aparece na vida cotidiana. Um exemplo ...

Abordagens de IA

Abordagem Simbólica:

- Representa o conhecimento por sentenças declarativas.
  - Deduz consequências por métodos de raciocínio lógico.

Exemplo:

$\forall x \forall y$ irmão(x, y) $\Rightarrow$ parente(x, y)

$\forall x \forall y \forall z$ pai(z, x) $\wedge$ pai(z, y) $\Rightarrow$ irmão(x, y)

pai(joão, maria)  Maria e João são parentes

pai(joão, eduardo)

- É necessário:
  - Identificar o conhecimento do domínio (modelo do problema).
  - Representá-lo utilizando uma linguagem formal de representação.
  - Implementar um mecanismo de inferência para utilizar esse conhecimento.

Abordagens de IA

Abordagem Não-Simbólica:

- Na abordagem Não-Simbólica, o conhecimento não é representado explicitamente por meio de símbolos, e sim, construído a partir de um processo de aprendizado, adaptação ou inferência.

Exemplos:

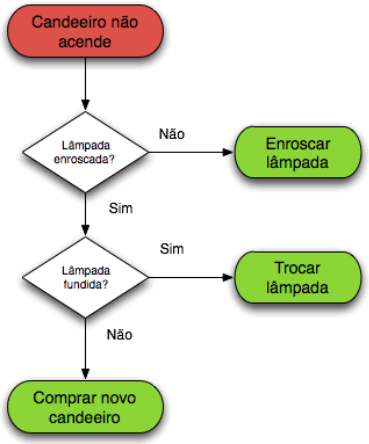
- Redes Neurais Artificiais, Computação Evolutiva, Sistemas Nebulosos (*Fuzzy*)... .

Problemas

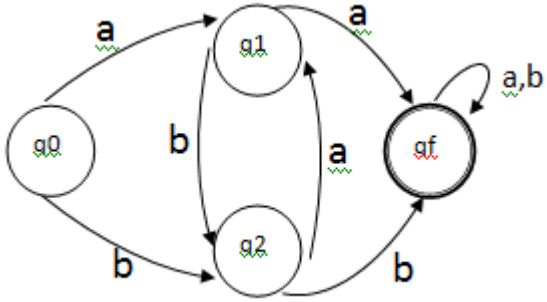
- Como saber qual é o melhor conjunto de ações para investir na bolsa?
- Quais são os melhores produtos para sugerir uma venda casada?
- Qual a melhor rota de entrega?
- Como determinar qual pessoa está entrando em casa?
- Como balancear o fluxo da rede conforme a variação do seu dia?
- Como diagnosticar um paciente?

Algoritmos

- **Algoritmo determinístico**: um único caminho para um resultado.



- **Algotitmo não-determinístico**: diversos caminhos para um resultado, diversos resultados corretos ou não ter resultado.



Áreas da IA

- Pesquisa operacional: otimização e busca heurística em geral.
- Processamento de linguagem natural: tradução automática, verificadores ortográficos e sintáticos, reconhecimento da fala.
- Visão Computacional: reconhecimento de padrões, processamento de imagens, realidade virtual.
- Sistemas especialistas: atividades que exigem conhecimento especializado e não formalizado:
  - Tarefas: diagnóstico, previsão, monitoramento, análise, planejamento, projeto, etc

Aplicações

- *Business Inteligence*
- Sistemas de segurança
- Reconhecimento de padrões
- Decisões médicas
- Robótica
- Avaliação de fluxo de redes
- Etc.

Aplicações

- Jogos e simulações

Eventos

- International Aerial Robotics Competition



- Robo Cup (SBC)



Eventos

- Google AI Challenge
- Machine Learning Challenges
- AIIDE Starcraft Competition
- Mario AI Championship

Exercícios

- **Inteligência Artificial pode ser definida como:**

a) Sub-área da ciência da computação que permite construir sistemas não-deterministicos para a resolução de problemas do mundo real.

b) Sub-área da ciência da computação que permite construir sistemas deterministicos para a resolução de problemas do mundo real.

c) Área da ciência da computação que permite implementar sistemas não-deterministicos para a resolução de problemas somente do mundo artificial com o uso de pouca complexidade computacional.

d) São sistemas de informação que se interagem com algum banco de dados.

e) São sistemas de computação que se interagem com algum bando de dados.

Exercícios

- **Na IA encontramos diversos problemas associados a inferir (raciocinar) conhecimento do mundo real com a finalidade de encontrar um conjunto de soluções optimas que satisfaçam os requisitos do problema. Estes problemas possuem uma caracteristica em comum, eles são não-deterministicos. O que representa o termo não-deterministico?**

a) Representa um algoritmo com apenas uma solução para o problema.

b) Significa que o algoritmo sempre determina uma solução para um problema.

c) Que não possui apenas uma solução para o problema, ou seja, pode possuir várias ou não ter solução.

d) O algoritmo sempre encontra uma solução optima.

e) O algoritmo sempre encontra uma solução deterministica para o problema.

Exercícios

- **Em sistemas de informação o "dado" se transforma em informação através de um processamento. Em IA a informação pode ser transformada em:**

a) Conhecimento humano.

b) Conhecimento geral.

c) Conhecimento específico acerca de apenas dois dominios de problema.

d) Conhecimento geral acerca de diversos dominios de problemas.

e) Conhecimento específico acerca de um dominio de problemas.

Exercícios

- **Inteligência Artifical resolve diversos problemas que ajudam o ser humano em suas tarefas cotidianas. Qual das alternativas abaixo é a correta?**

a) Sistema de freios comum, Comando de voz no celular e ao ligar o carro, Detector de faces e sorrisos em cameras digitais.

b) Sistema de freios ABS, Comando de voz no celular e ao ligar o carro, Detector de faces e sorrisos em cameras digitais.

c) Sistema de freios comum, modelagem do ambiente físico e comportamento dos personagens em jogos de computadores.

d) Navegação de robôs de forma segura e efeciente, reconhecimento de iris, tirar fotografias.

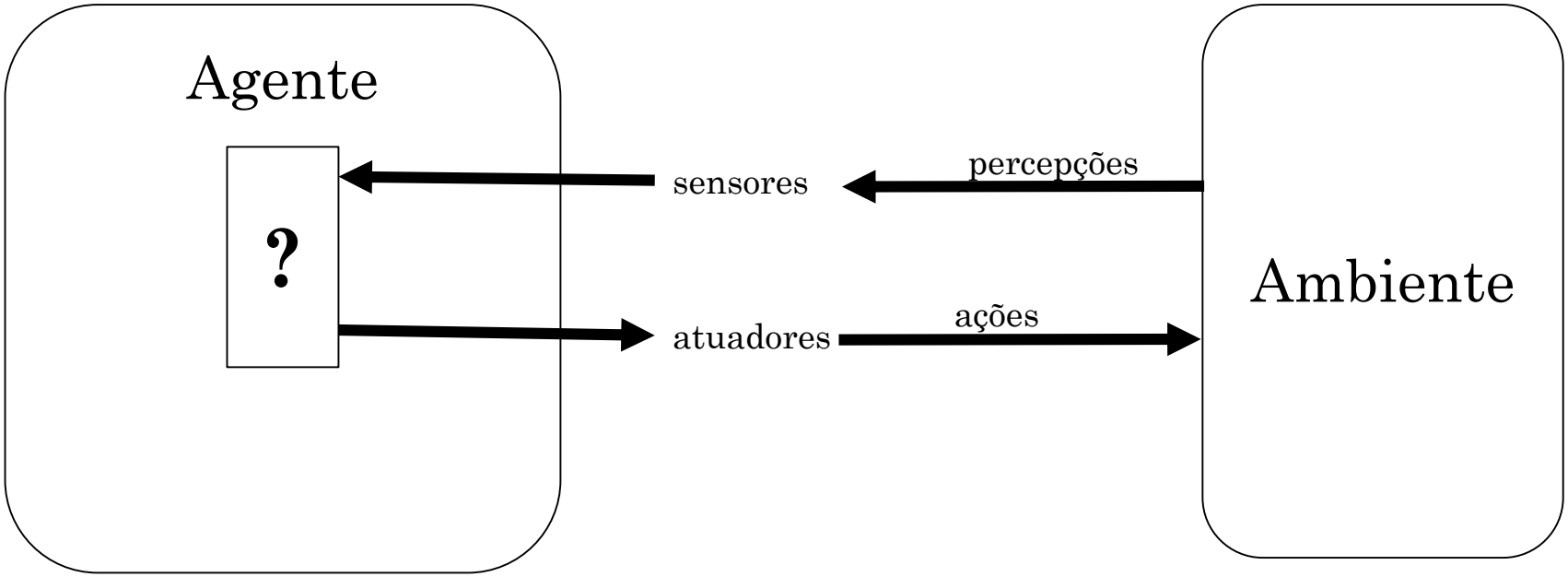
e) Localização de informações relevantes através dos mecanismos de busca dos sites de pesquisa, trocar canais em uma TV.

Agentes Inteligentes

Edirlei Soares de Lima (PUC RIO) / Prof. Dr. Silvio do Lago Pereira (IME)

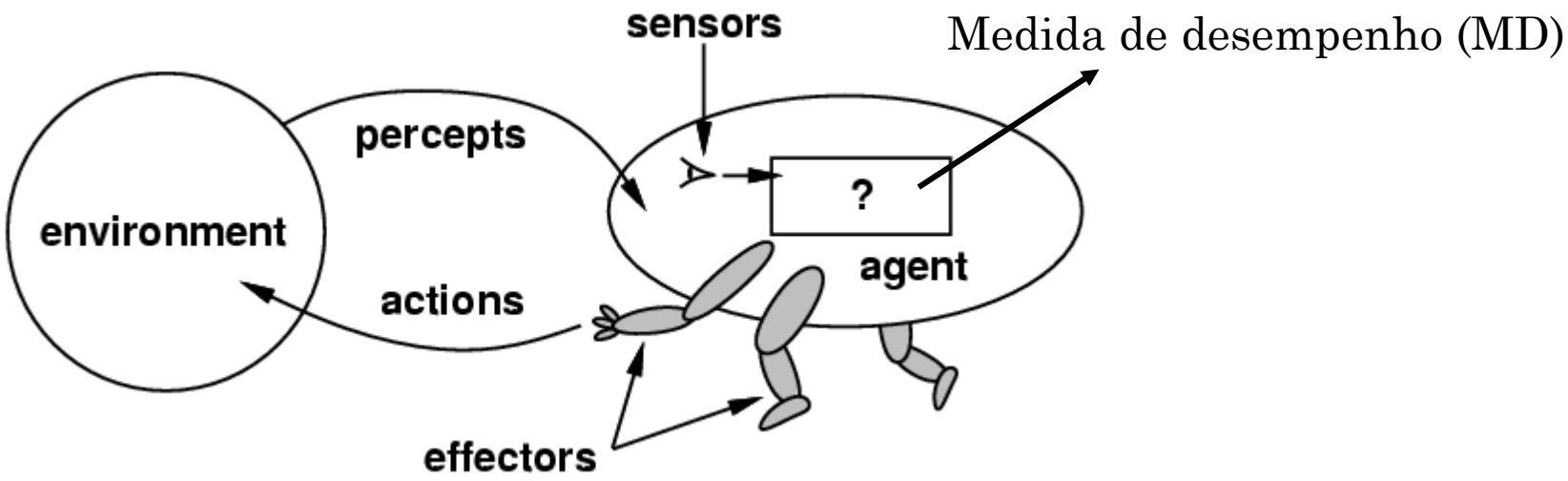
Definição

- Um agente é algo capaz de perceber seu ambiente por meio de sensores e de agir sobre esse ambiente por meio de atuadores.



Agentes Inteligentes

- Agentes são diferente dos programas comuns pois operam sob controle autônomo, percebem seu ambiente, adaptam-se a mudanças e são capazes de assumir metas.



Exemplos

- **Agente humano**
  - Sensores: Olhos, ouvidos e outros órgãos.
  - Atuadores: Mãos, pernas, boca e outras partes do corpo.

- **Agente robótico**
  - Sensores: câmeras e outros sensores.
  - Atuadores: vários motores.

- **Agente de software**
  - Sensores: entrada do teclado, conteúdo de arquivos e pacotes vindos da rede.
  - Atuadores: tela, disco rígido, envio de pacotes pela rede.

Exemplos de agentes

Tipo de agente	Medida de desempenho	Ambiente	Atuadores	Sensores
Sistema de diagnóstico médico	Paciente saudável, minimizar custos, processos judiciais	Paciente, hospital, equipe	Exibir perguntas, testes, diagnósticos, tratamentos, indicações	Entrada pelo teclado para sintomas, descobertas, respostas do paciente
Sistema de análise de imagens de satélite	Definição correta da categoria da imagem	Link de transmissão de satélite em órbita	Exibir a categorização da cena	Arrays de pixels em cores
Robô de seleção de peças	Porcentagem de peças em bandejas corretas	Correia transportadora com peças; bandejas	Braço e mão articulados	Câmera, sensores angulares articulados
Controlador de refinaria	Maximizar pureza, rendimento, segurança	Refinaria, operadores	Válvulas, bombas, aquecedores, mostradores	Sensores de temperatura; pressão; produtos quimicos quimicos
Instrutor de inglês interativo	Maximizar nota de aluno em teste	Conjunto de alunos, testes de agência	Exibir exercícios, sugestões, correções	Entrada pelo teclado

Mapeamento das percepções em ações

- O comportamento de um agente é dado abstratamente pela **função do agente**:

  f = P → A

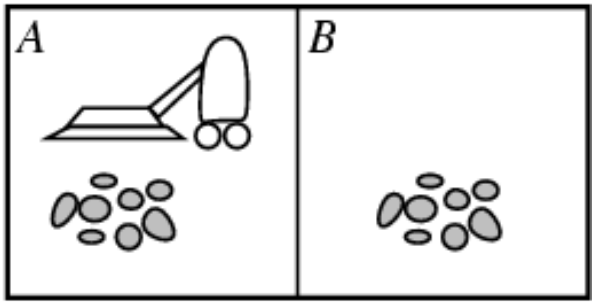
  onde é a **P** é uma sequência de percepções e **A** é uma ação.

- **Sequência de percepções:** histórico completo de tudo que o agente percebeu.

Exemplo (Aspirador de Pó)

- **Percepções:** Local e conteúdo.
  - Exemplo: [A, sujo]

- **Ações:** Esquerda, Direita, Aspirar.



Exemplo

Sequência de Percepções	Ação
[A, Limpo]	Direita
[B, Sujo]	Aspirar
[B, Limpo]	Esquerda
[A, Sujo]	Aspirar

Comportamento do Agente: Se o quadrado atual estiver sujo, então aspirar, caso contrário mover para o outro lado.

Medidas de Desempenho (MD)

- O agente deve sempre executar a ação "**correta**" baseado no que ele percebe para ter sucesso.
  - O conceito de sucesso do agente depende uma **medida de desempenho** objetiva.
    - **Exemplos:** quantidade de sujeira aspirada, gasto de energia, gasto de tempo, quantidade de barulho gerado, etc.
  - A medida de desempenho deve refletir o resultado realmente desejado.

Agente Racional

- Para cada sequência de percepções possíveis deve-se selecionar uma ação que espera-se que venha a maximizar sua medida de desempenho, dada a evidência fornecida pela sequência de percepções e por qualquer conhecimento interno do agente.

- Qual seria a medida de desempenho ideal para o agente aspirador de pó racional?

Agentes Racionais

- Os agentes podem executar ações para coleta de informações.
  - Um tipo importante de coleta de informação é a exploração de um ambiente desconhecido.
- Os agentes também podem aprender, ou seja, modificar seu comportamento dependendo do que ele percebe ao longo do tempo.
  - Nesse caso o agente é chamado de autônomo.
  - Um agente que aprende pode ter sucesso em uma ampla variedade de ambientes.

Modelagem de um Agente

- O processo de modelagem de um agente envolve a definição de:
  - Medida de Desempenho
  - Ambiente
  - Atuadores
  - Sensores

Exemplo

- **Medida de desempenho:** viagem segura, rápida, sem violações às leis de trânsito, confortável para os passageiros, maximizando os lucros.
- **Ambiente:** ruas, estradas, outros veículos, pedestres, clientes.
- **Atuadores:** direção, acelerador, freio, embreagem, marcha, seta, buzina.
- **Sensores:** câmera, sonar, velocímetro, GPS, acelerômetro, sensores do motor, teclado ou microfone.

Propriedades do Ambientes

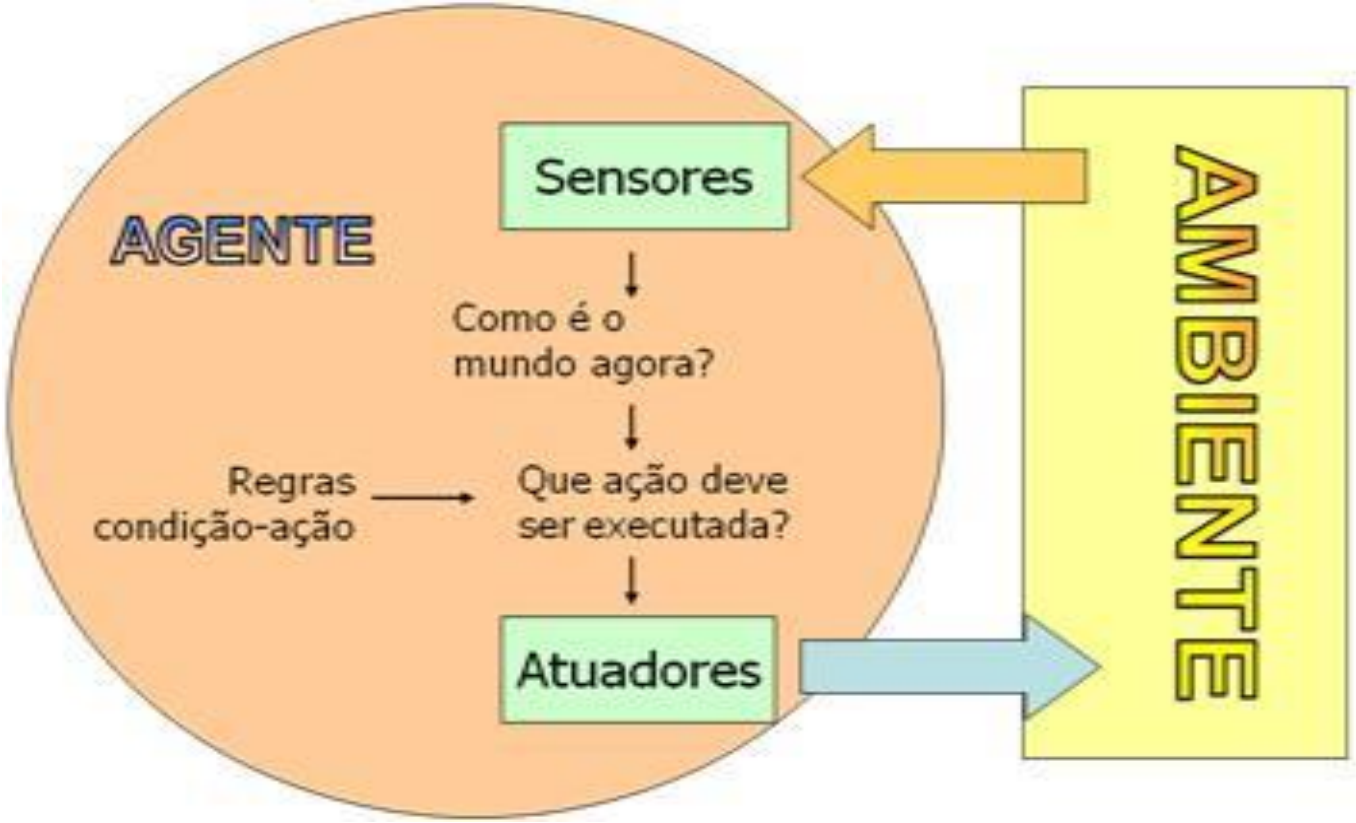
- **Determinístico:**
  - O próximo estado do ambiente é completamente determinado pelo estado atual e pela ação executada pelo agente.
- **Não-Determinístico:**
  - O próximo estado do ambiente é desconhecido. Não se tem certeza do que pode acontecer com o ambiente ao executar uma ação.
- **Estático:**
  - O ambiente não muda enquanto o agente pensa.
- **Dinâmico:**
  - O ambiente pode mudar enquanto o agente pensa ou está executando uma ação.

Propriedades do Ambientes

- **Discreto:**
  - Um número limitado e claramente definido de percepções, ações e estados.
- **Contínuo:**
  - Um número possivelmente infinito de percepções, ações e estados.
- **Agente Único:**
  - Um único agente operando sozinho no ambiente.
- **Multi-Agente**
  - Vários agentes interagindo ambiente.
  - Multi-agente cooperativo
  - Multi-agente competitivo

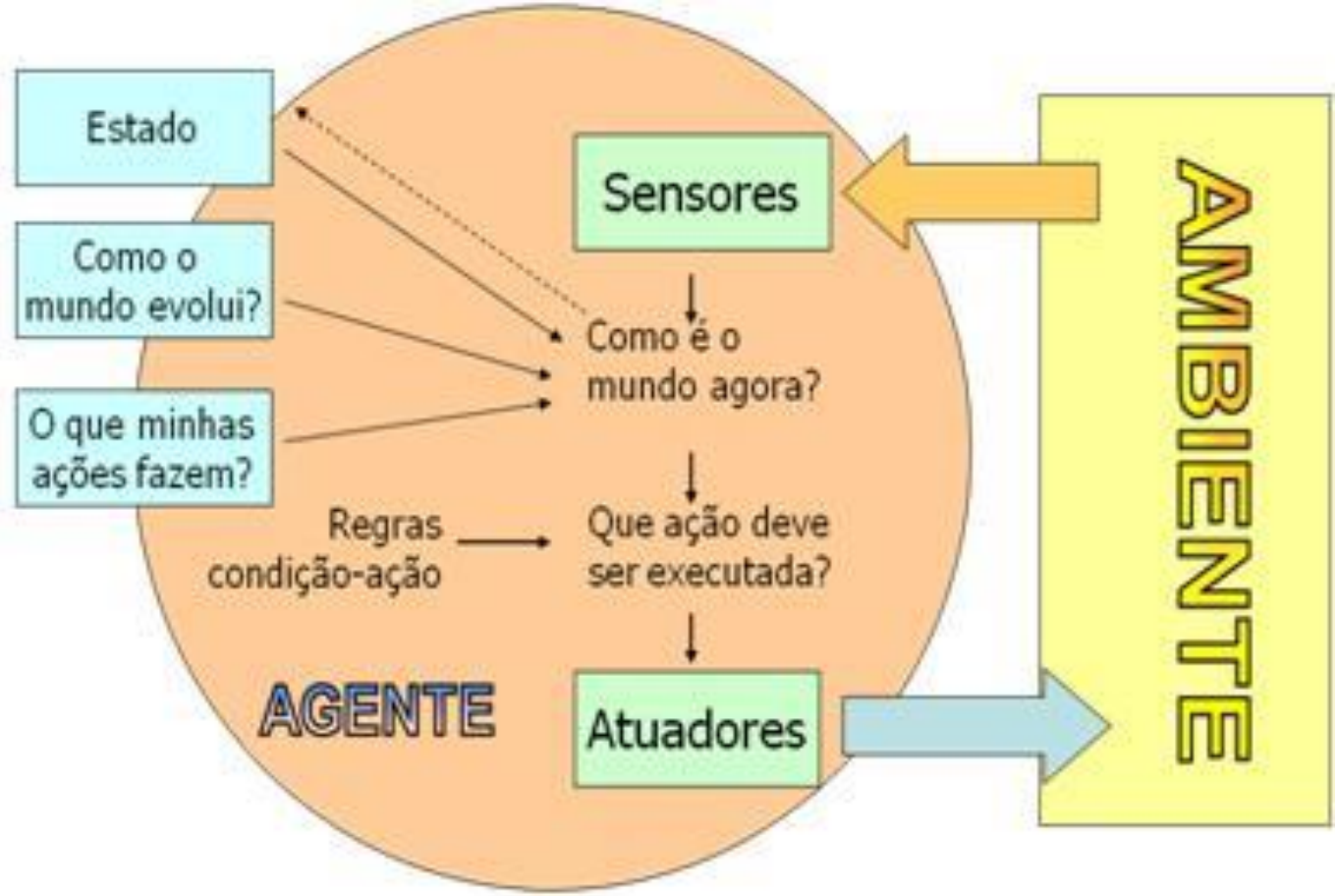
Tipos Básicos de Agentes

- Agentes reativos simples.



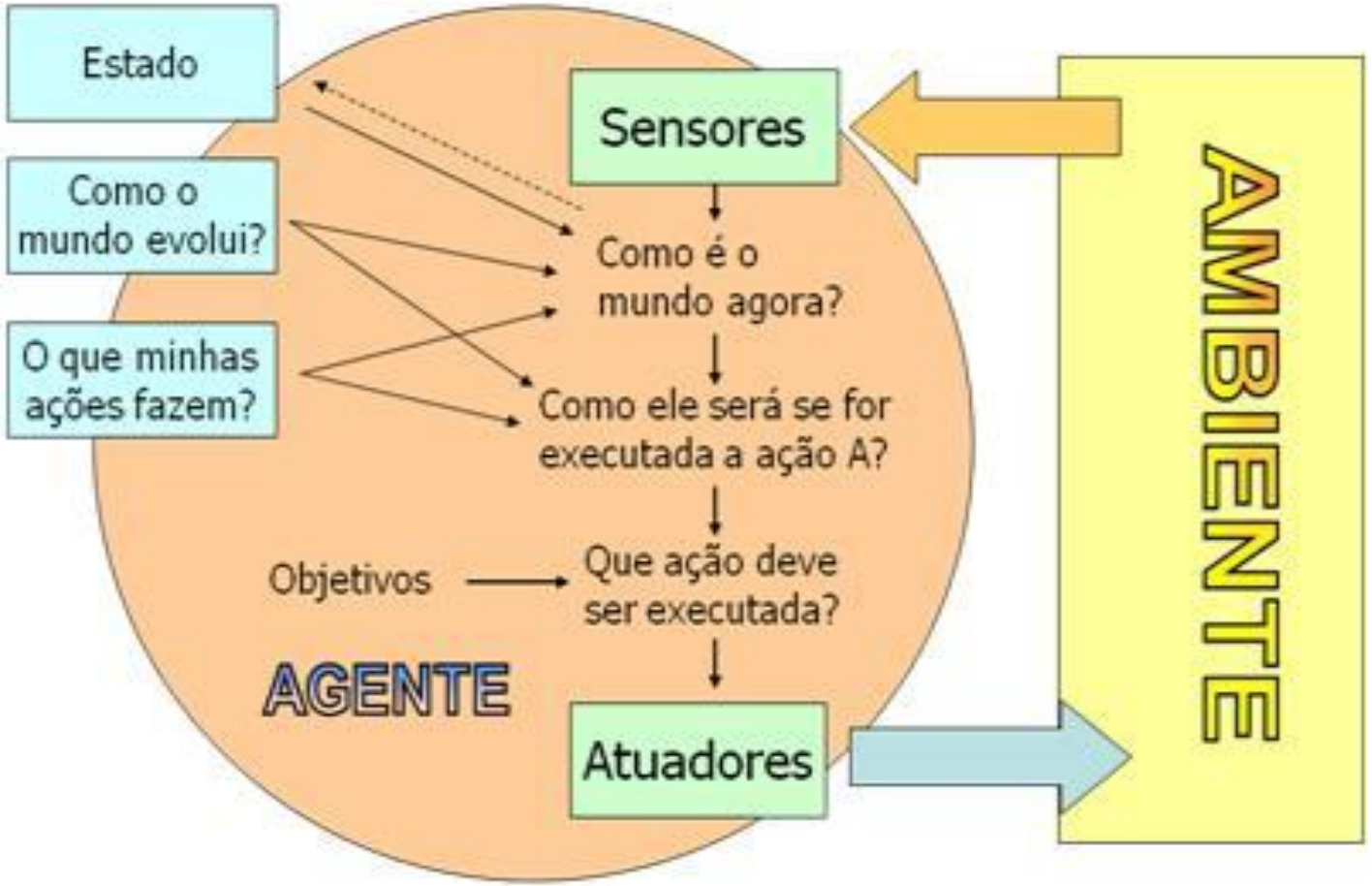
Tipos Básicos de Agentes

- Agentes reativos baseados em modelos.



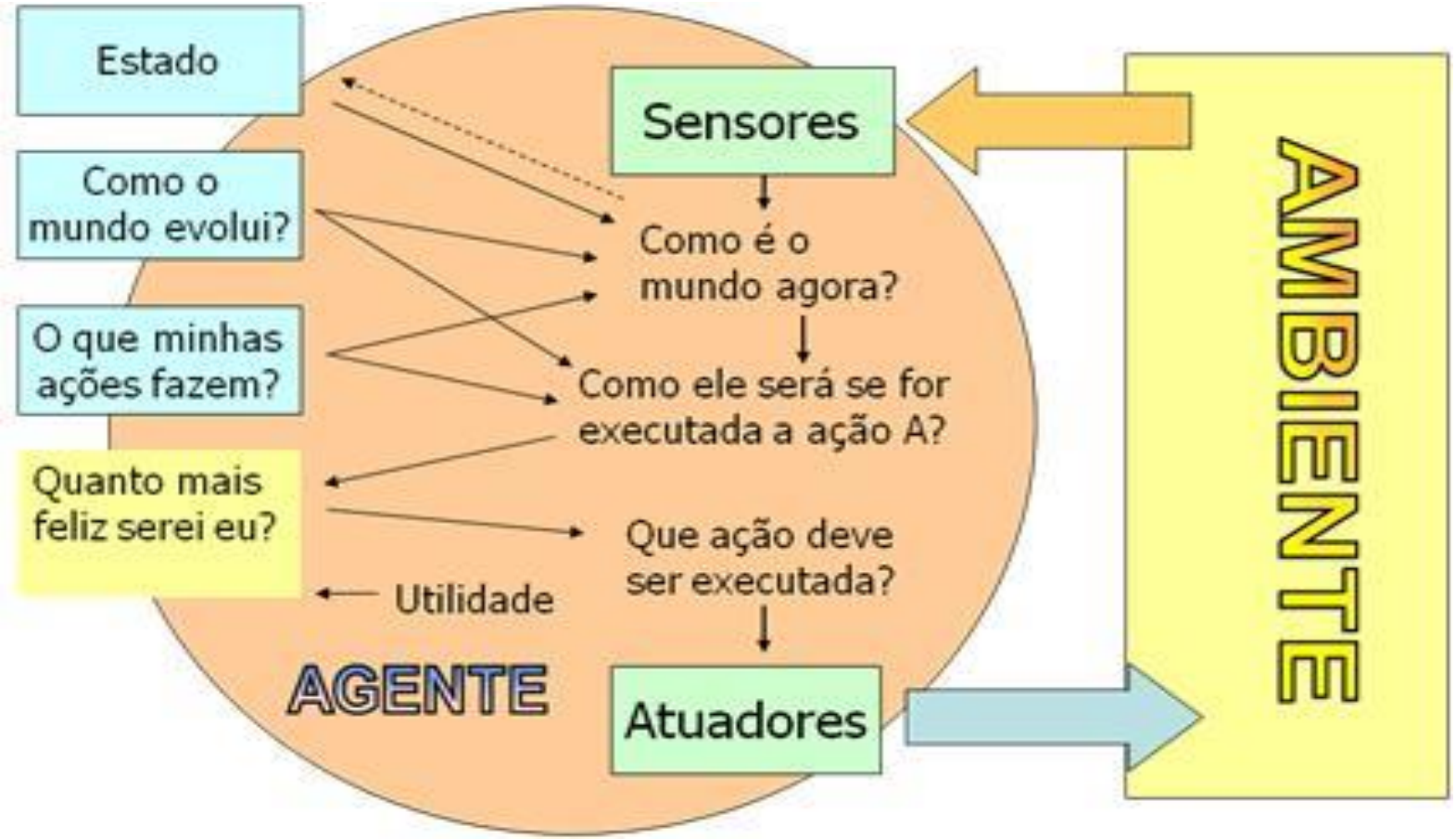
Tipos Básicos de Agentes

- Agentes Baseados em Objetivos



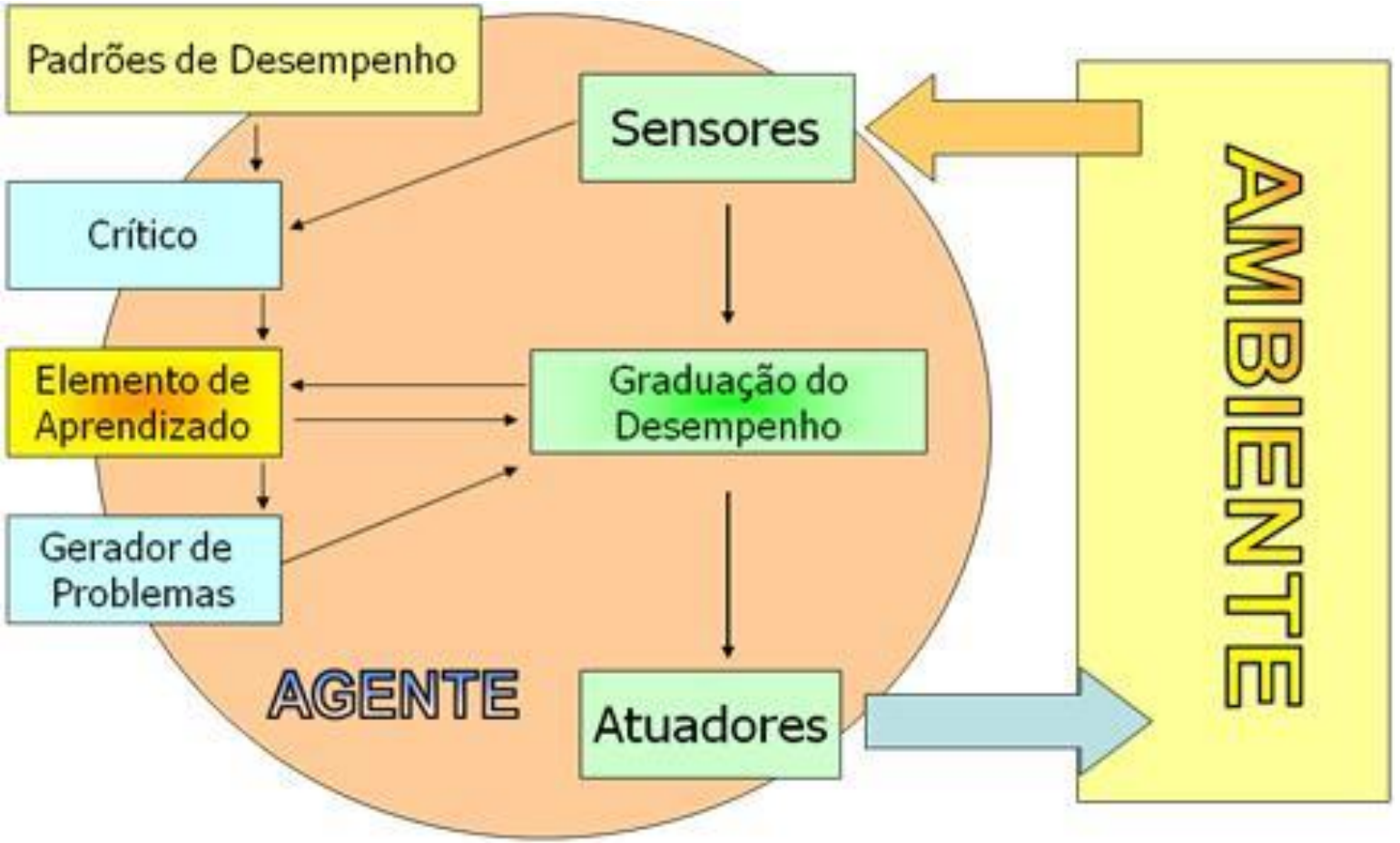
Tipos Básicos de Agentes

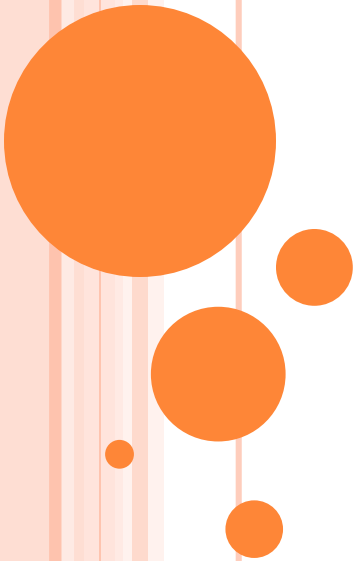
- Agentes Baseados em Utilidades



Tipos Básicos de Agentes

- Agentes com Aprendizagem



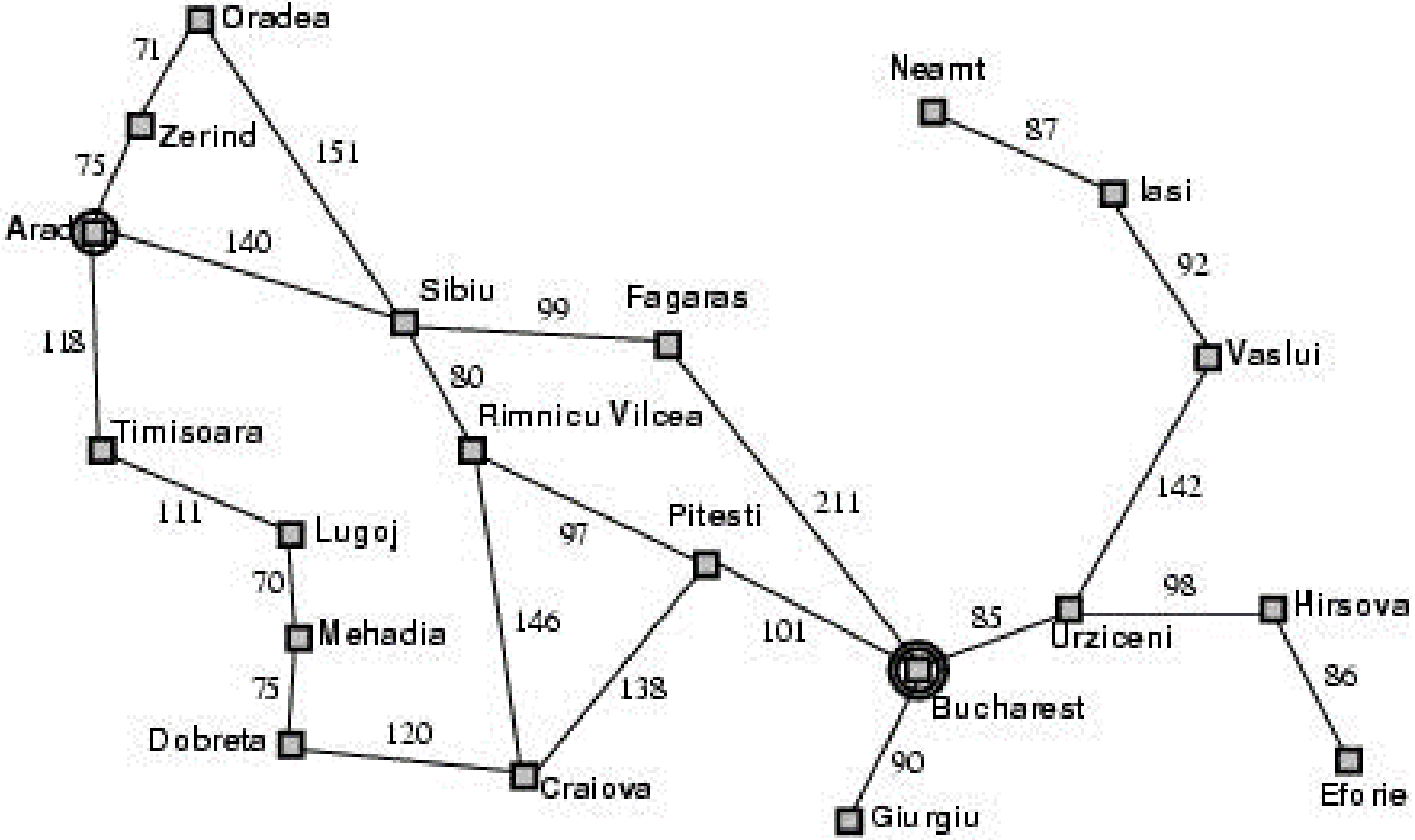


Espaço de Estados

Definição

- A resolução de problemas inicia em construir e pesquisar o Espaço de Estados do problema;
- Espaço de Estados:
  - é uma forma de representar conhecimento sob um domínio específico de problemas;
  - é o conjunto de todos os estados pesquisados a partir de um estado inicial;
  - forma um grafo onde os **nós são estados** e os **arcos entre os nós são ações** (executando as ações, há transição de um estado para o outro).
- O caminho (*path*) do espaço de estados é a sequência de estados que são conectados por uma sequencia de ações.

ESPAÇO DE ESTADOS POR MEIO DE GRAFOS



Forma de Representação

- (Ordea,Zerind,71) (Ordea,Sibiu,151)
- Façam o restante...

Problemas

- Qual é o melhor caminho para chegar a Eforie?
- Qual é o melhor para chegar a Vaslui passando por Piteste e Sibiu?
- E se contássemos com pedágios?
- Como verificar o caminho com menos pedágios?
- E se colocássemos o nível de periculosidade nas estradas (números de acidentes)?
- Como calcular o caminho mais seguro?

Resolução dos problemas

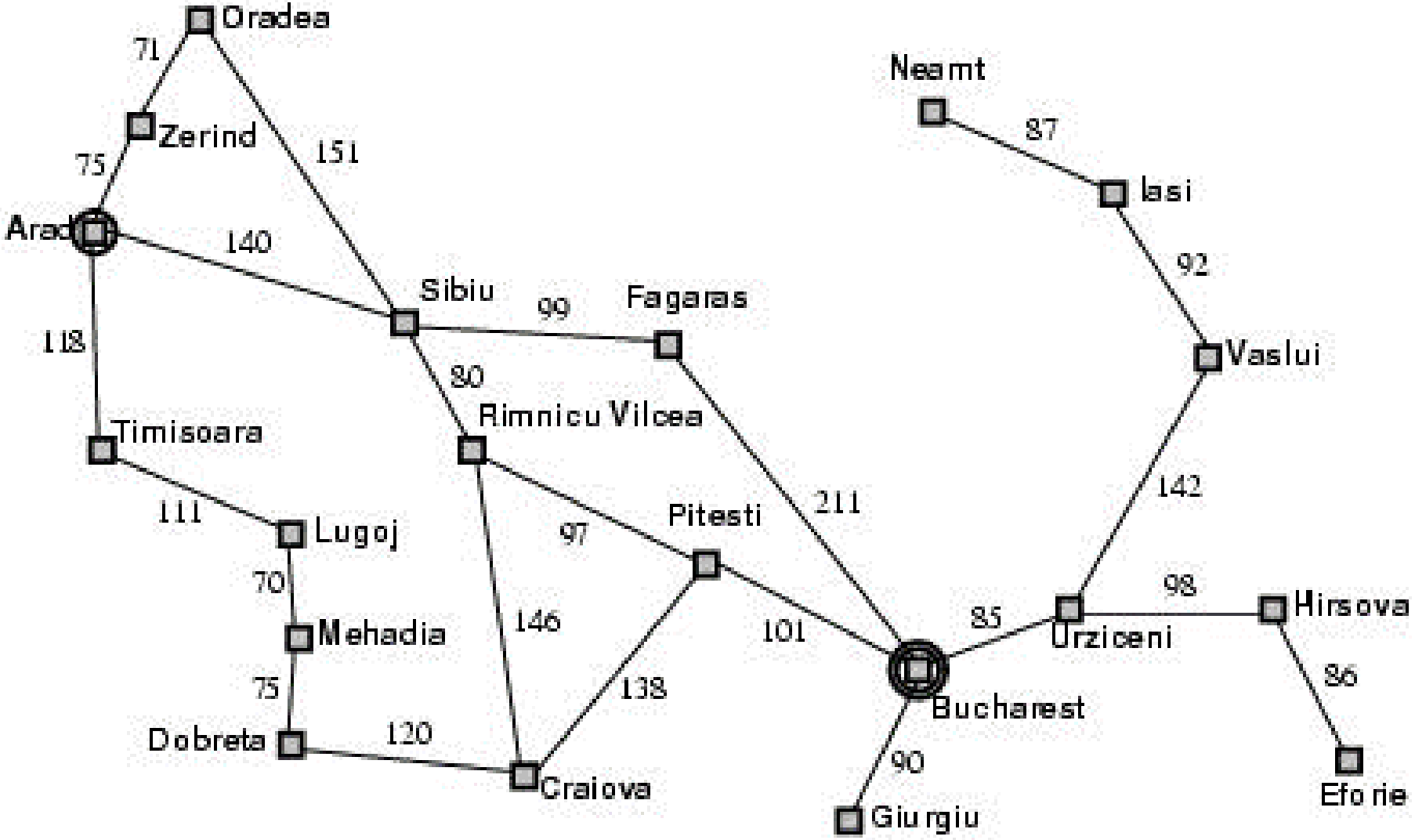
- Resolução de problemas (*problem-solving*) pode ser construido por agentes de software a partir de uma descrição formar de problemas que decidem o que fazer para encontrar uma sequencia de ações baseadas em técnicas de buscas que leva de um estado inicial aos estados objetivos em um Espaço de estados;
- Resolução de Problemas = Descrição formal do problema + Escolha da Estratégia de busca (ou politica de busca);

Resolução dos problemas

Exemplo:

Um agente de resolução de problemas possui como espaço de estados o mapa da Romenia, conforme visto no conteúdo anterior. Assim, o agente pode ir de ARAD até BUSCHAREST analisando diferentes sequências possíveis de ações que levam para os estados de valores conhecimentos, com o intuito de escolher a melhor sequência, ou seja, realiza uma pesquisa (ou busca) no espaço de estados.

ESPAÇO DE ESTADOS POR MEIO DE GRAFOS



Resolução dos problemas

- Um algoritmo de pesquisa em um Espaço de estados captura um problema como uma entrada e retorno a solução na forma de uma sequencia de ações.
- Para resolver problemas com o uso de espaço de estados é preciso definir buscas em uma Árvore Binária de Pesquisa (*Binary Search Tree*);

Pesquisa (Busca) no Espaço de Estados

- Busca é um mecanismo geral de pesquisa em árvore;
- Uma busca inicia-se na escolha de um estado inicial. Após aplica uma sequência de ações com o intuito de se construir um ramo na árvore que conduz a um estado objetivo;
- Existem diversas buscas no espaço de estados, mas as buscas clássicas são: Largura (*Breadth-First Search*) e a Profundidade (*Depth-First Search*);
- Elas são conhecidas como **buscas cegas**, pois não são guiadas por conhecimento.

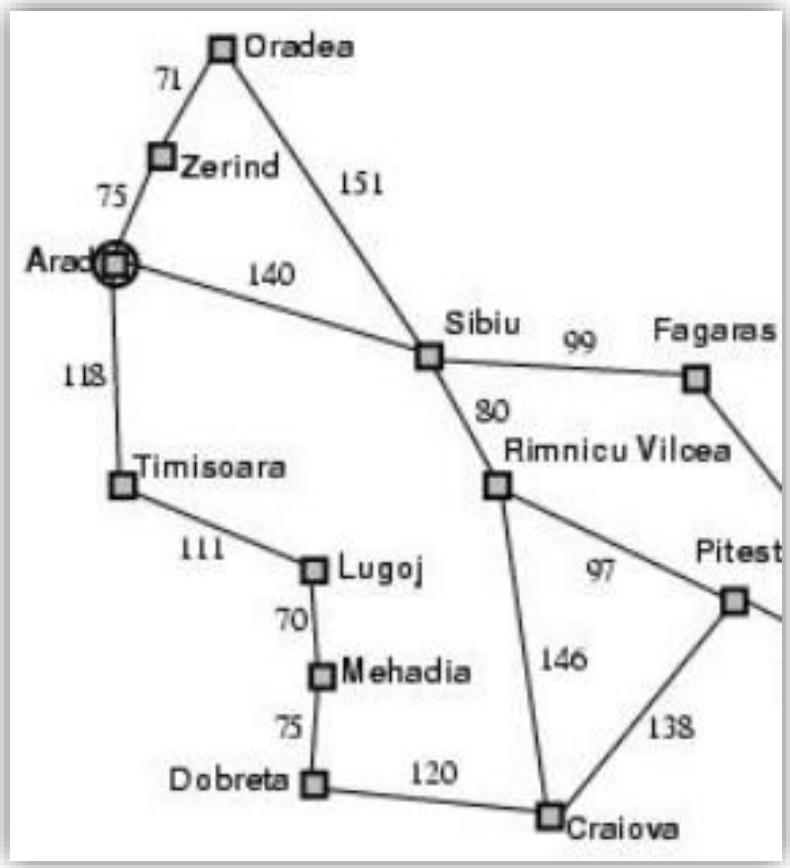
Exercícios - 1

- Espaço de estados pode representar as "regras do jogo" para um dado problema. Porém sua representação pode mudar sendo dependente de:

a) Uma linguagem de programação
b) Uma Interface gráfica.
c) Regras de um banco de dados.
d) Lógicas para a resolução de problemas formais.

Exercícios - 2

- **Tendo como base o grafo abaixo, como poderia ser a representação do conhecimento indo de ARAD (é o estado inicial) para ORADEA (estado objetivo), levando em consideração a informação: kilometragem?**



Mapa Simplificado das rodovias da Romenia

Fonte: RUSSEL Stuart; NORVING Peter. Artifical Intelligence: A modern Approach. 2.Edition. New Jersey: Prentice Hall, 2003. Pg. 63.

a) (arad, zerind, 146) (zerind, arad, 146)

b) (zerind, zerind, 71) (arad, zerind, 75)

c) (arad, zerind) (zerind, arad)

d) (arad, zerind, 75) (zerind,ordea, 71)

e) (arad, zerind, 75) (zerind, arad, 71)

Exercícios - 3

- **A representação do conhecimento por espaço de estados é um modelo que pode ser implementado por uma estrutura de dados. Escolha a alternativa correta para informar qual estrutura de dados utilizada.**

a) Tabela Hash.

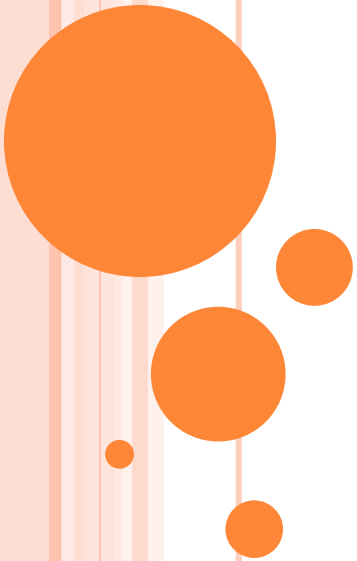
b) Árvore Binária.

c) Pilha e Fila.

d) Orientação a Objetos.

Sugestão de Leitura

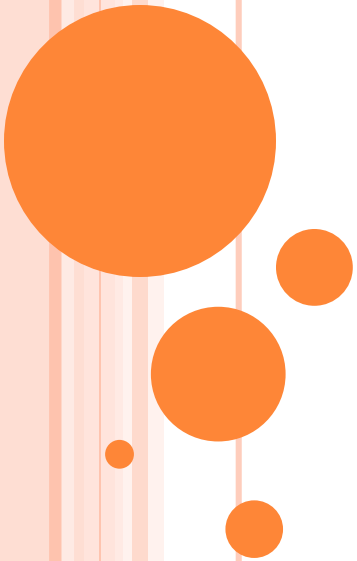
- Artigo: “Busca no Espaço de Estados” – Prof. Dr. Silvio do Lago Pereira
- http://www.ime.usp.br/~slago/IA-Busca.pdf



Backtracking

Backtracking

- É a forma de como se pode resolver problemas.
- Uma regra é selecionada e, se não houver solução, os passos são “esquecidos” e uma outra regra é selecionada até que apresente sucesso, falha ou caminho sem saída.



Buscas em Árvore

Problema de Busca

- **Objetivo:** Conjunto de estados que satisfazem o objetivo.
- **Tarefa de Busca:** Encontrar a sequencia de ações que leva do estado atual até um estado objetivo.

- Quais são os estados?
- Quais são as ações?
- Nível de abstração?

Problema de Busca

- O processo de tentar encontrar uma sequencia de ações que leva de um estado até um estado objetivo é chamado de busca.

- Uma vez encontrada a solução, o agente pode executar a sequência de ações para chegar no objetivo.

- Fases:
  - Formular objetivo
  - Buscar objetivo
  - Executar sequencia de ações

Definição do Problema

- A definição do problema é a primeira e mais importante etapa do processo de resolução de problemas de inteligência artificial por meio de buscas.

- Consiste em analisar o espaço de possibilidades de resolução do problema, encontrar sequências de ações que levem a um objetivo desejado.

Definição de um Problema

- **Estado Inicial:** Estado inicial do agente.
  - Ex: Em(Arad)
- **Estado Final:** Estado buscado pelo agente.
  - Ex: Em(Bucharest)
- **Ações Possíveis:** Conjunto de ações que o agente pode executar.
  - Ex: Ir(Cidade, PróximaCidade)
- **Espaço de Estados:** Conjunto de estados que podem ser atingidos a partir do estado inicial.
  - Ex: Mapa da Romênia.
- **Custo**: Custo numérico de cada caminho.
  - Ex: Distância em KM entre as cidades.

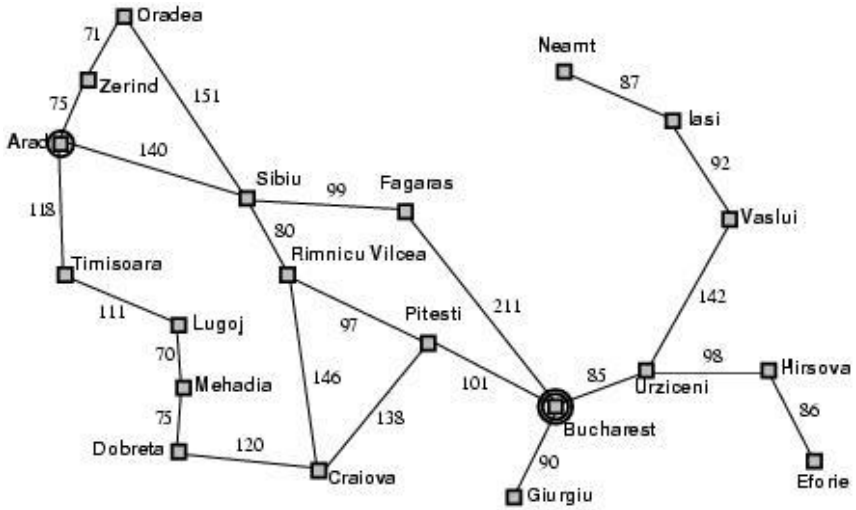
Como Encontrar a Solução?

- Uma vez o problema bem formulado, o estado final (objetivo) deve ser "buscado" no espaço de estados.
- A busca é representada em uma árvore de busca:
  - Raiz: corresponde ao estado inicial;
  - Expande-se o estado corrente, gerando um novo conjunto de sucessores;
  - Escolhe-se o próximo estado a expandir seguindo uma **estratégia de busca;**
  - Prossegue-se até chegar ao estado final (solução) ou falhar na busca pela solução;

Buscando Soluções

- O espaço de estados é diferente da árvore de buscas.

  Exemplo:



  - 20 estados no espaço de estados (20 cidades);
  - Número de caminhos infinito;
  - Árvore com infinitos nós;

Código Descritivo – Busca em Árvore

Função BuscaEmArvore(Problema, Estratégia) **retorna** solução ou falha

INICIO

Inicializa a arvore usando o estado inicial do Problema

LOOP FAÇA

SE não existem candidatos para serem expandidos

ENTÃO RETORNA falha

Escolhe um nó folha para ser expandido de acordo com a *Estratégia*

SE o nó possuir o estado final

ENTÃO RETORNA solução correspondente

SENÃO expande o nó e adiciona os nós resultantes a àrvore de busca

FIM LOOP

FIM

Métodos de Busca

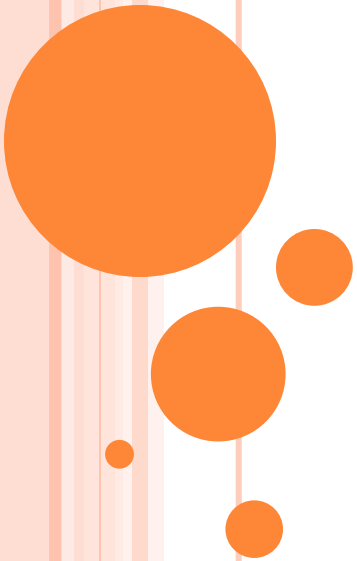
- **Busca Cega ou Exaustiva:** Não sabe qual o melhor nó da fronteira a ser expandido. Apenas distingue o estado objetivo dos não objetivos.

- **Busca Heurística:** Estima qual o melhor nó da fronteira a ser expandido com base em funções heurísticas.

- **Busca Local:** Operam em um único estado e movem-se para a vizinhança deste estado.

Busca Cega

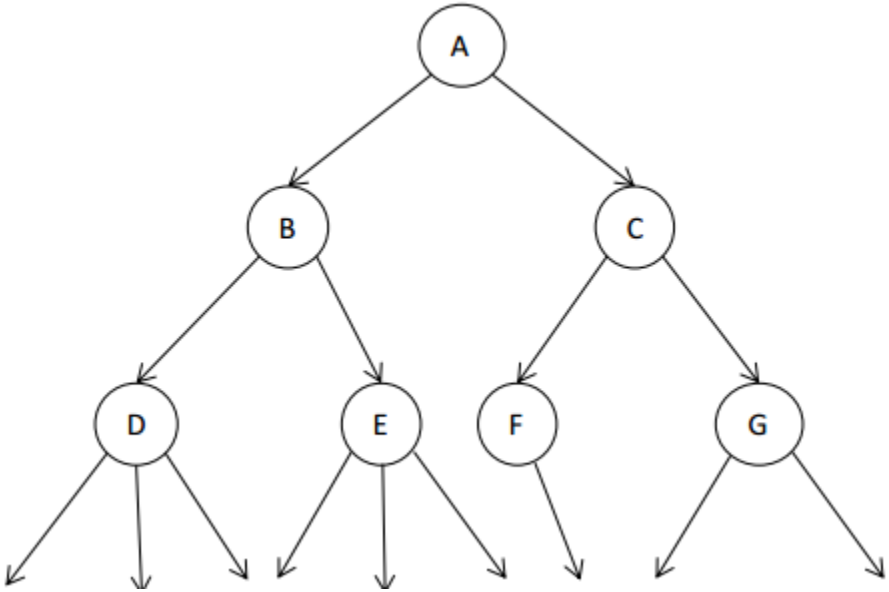
- Algoritmos de Busca Cega:
  - Busca em Largura;
  - Busca em Profundidade e Busca em profundidade Limitada;
  - Busca em Profundidade Iterativo;



Busca em Largura / Amplitude / *Breadth-First Search* (BFS)

Estratégia

- O nó raiz é expandido, em seguida todos os nós sucessores são expandidos, então todos próximos nós sucessores são expandidos, e assim em diante.

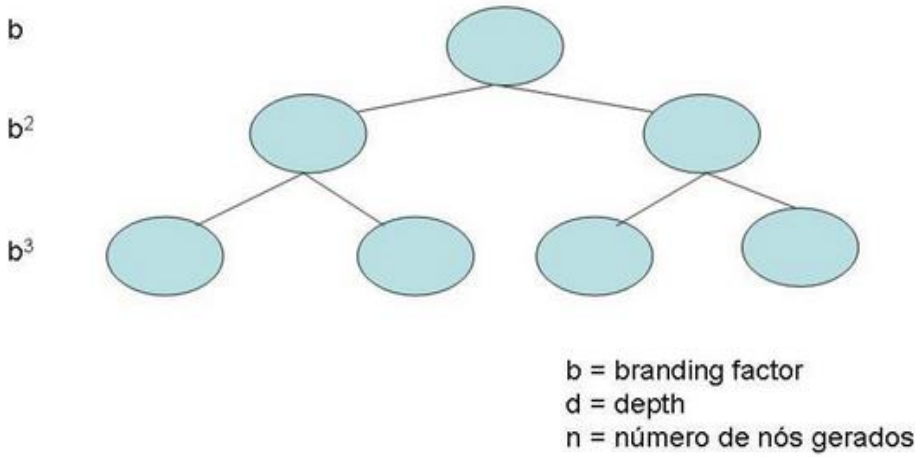


Busca em Largura

- Pode ser implementado com base no pseudocódigo da função “BuscaEmArvore” apresentado anteriormente. Utiliza-se uma estrutura de fila (*first-in-first-out*) para armazenar os nós das fronteira.

Complexidade

- $O(b^{d+1})$
  - d = profundidade do estado meta
  - b = fator de ramos

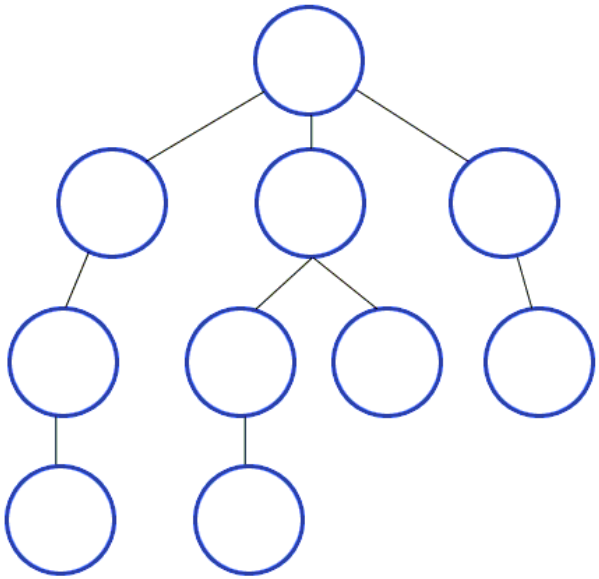


Profundidade (*d*)	Nós	Tempo	Memória
2	1100	0.11 ms	107 KB
4	111,100	11 ms	10.6 MB
6	10^7	1.1 seg	1 GB
8	10^9	2 min	103 GB
10	10^{11}	3 horas	10 TB
12	10^{13}	13 dias	1 PB
14	10^{15}	3.5 anos	99 PB

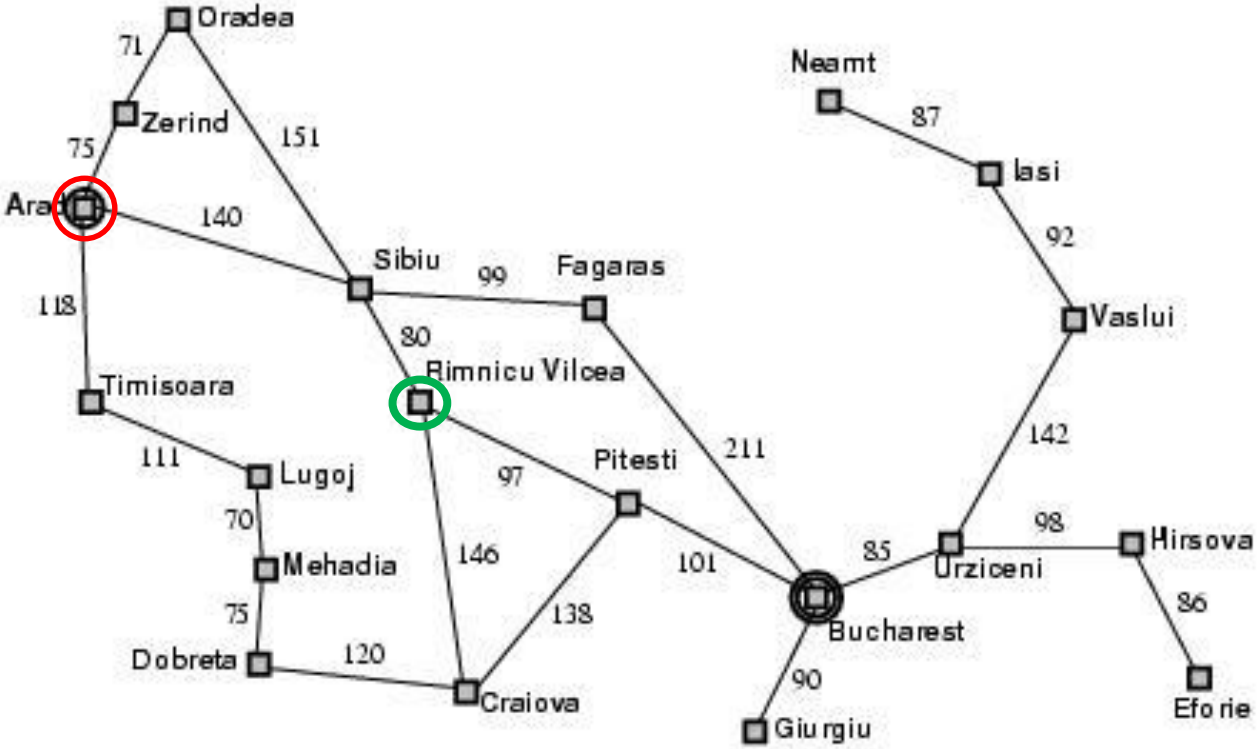
** Considerando o número de folhas b = 10 e cada nó ocupando 1KB de memória.*

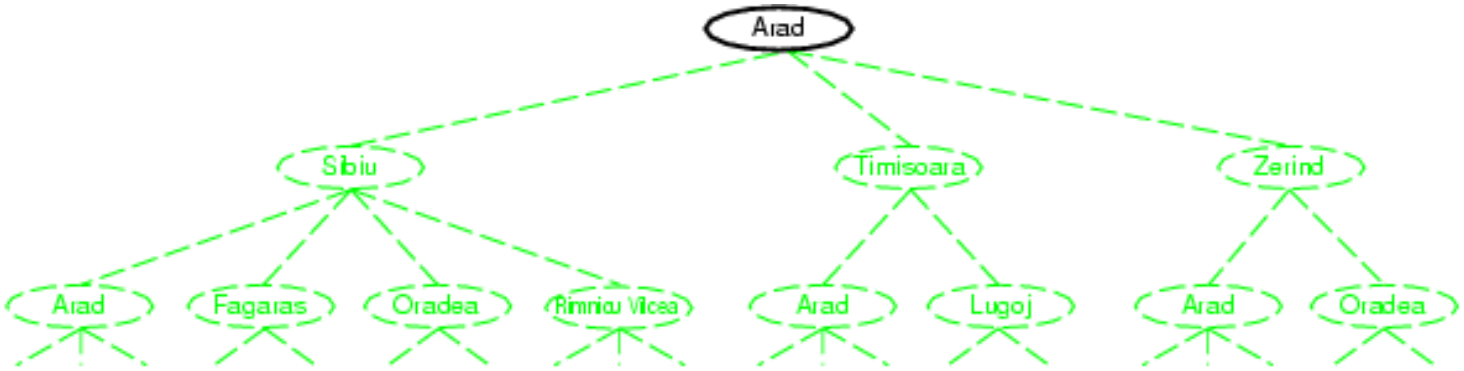
$n = b + b^2 + b^3 + \ldots + (b^{d+1} - b) = O(b^{d+1})$

Algoritmo

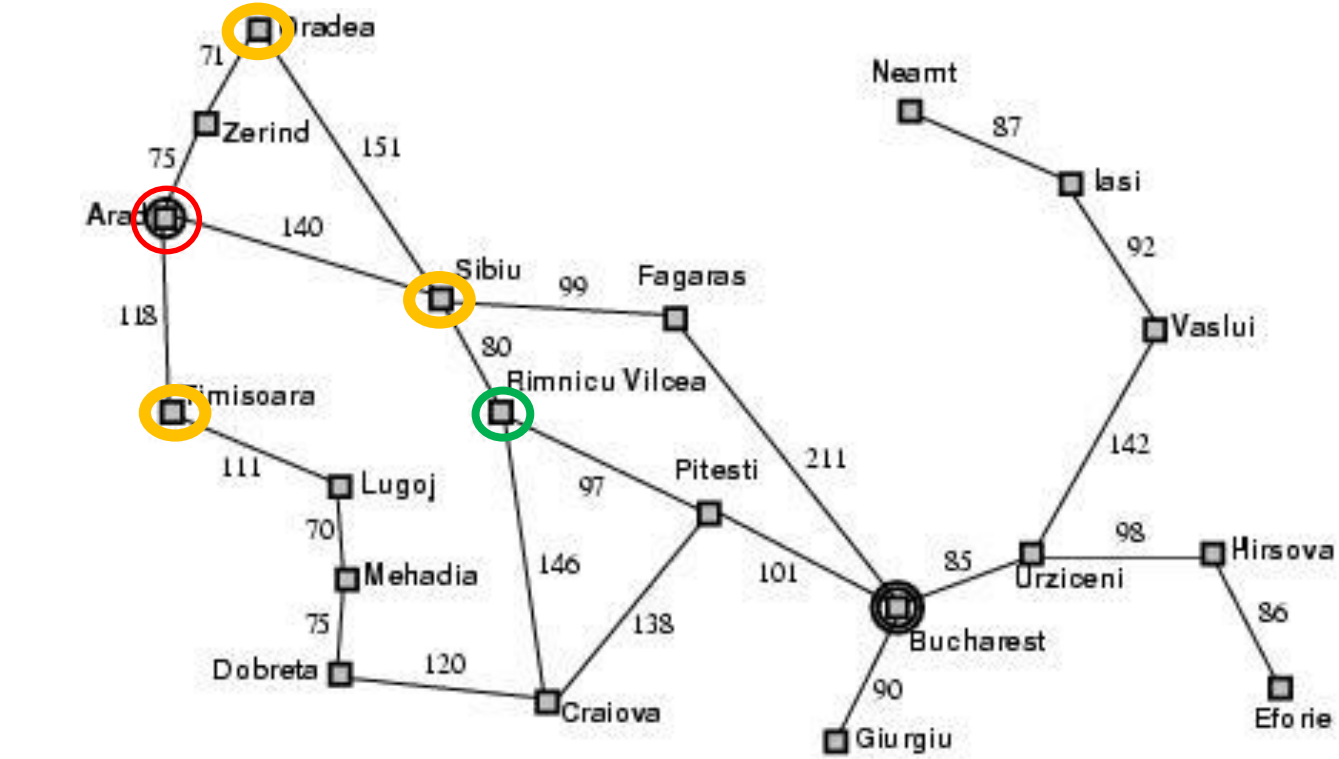


Mapa do problema

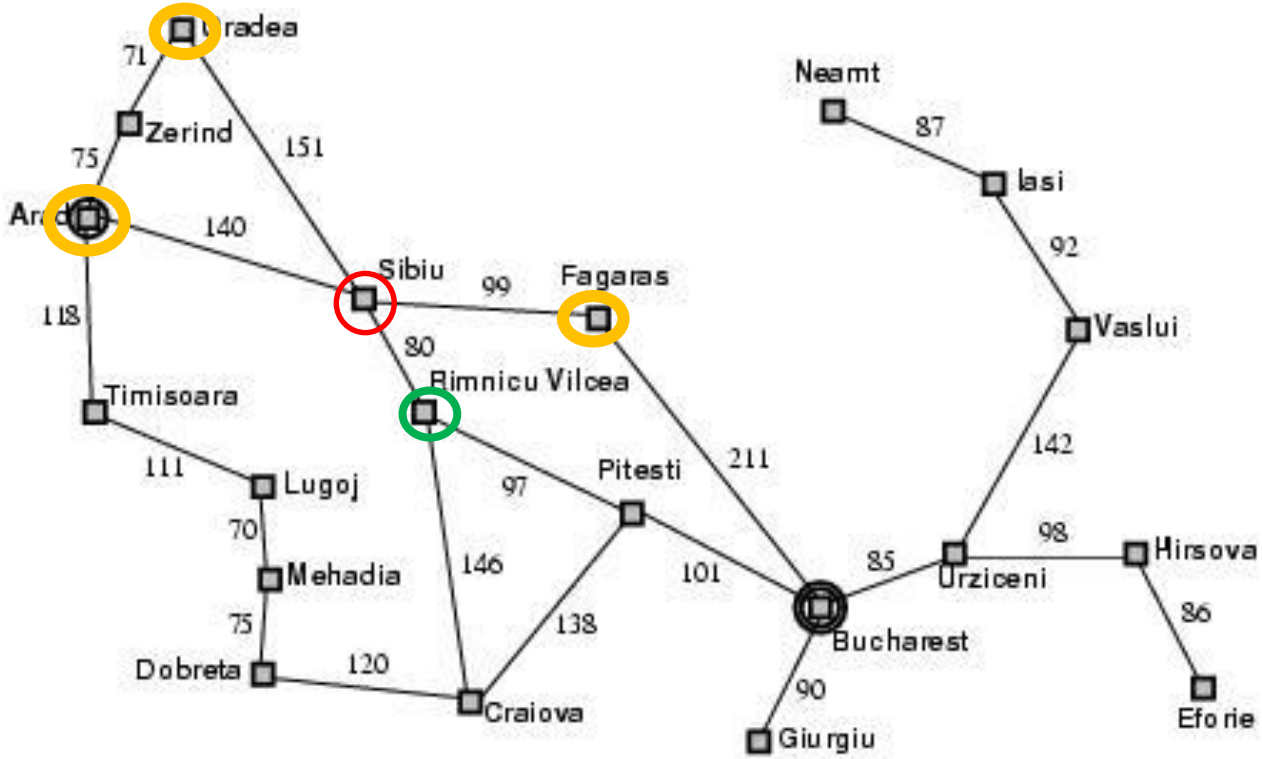


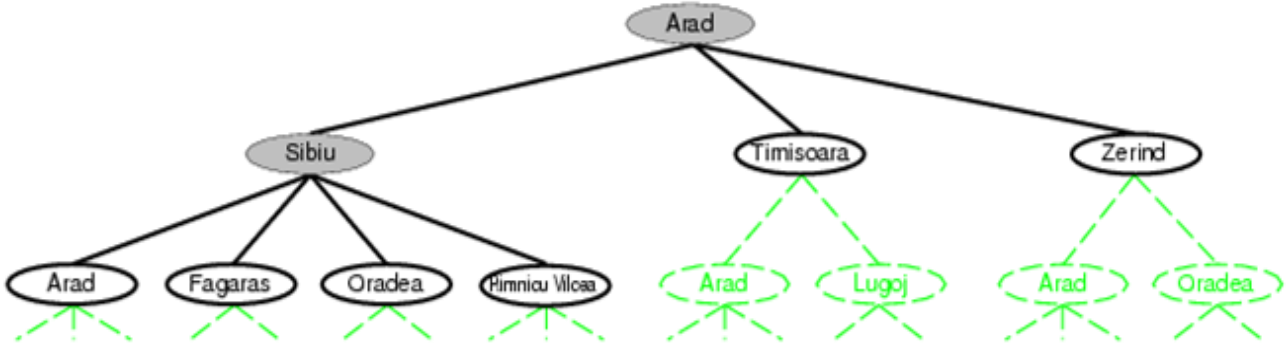


Mapa do problema



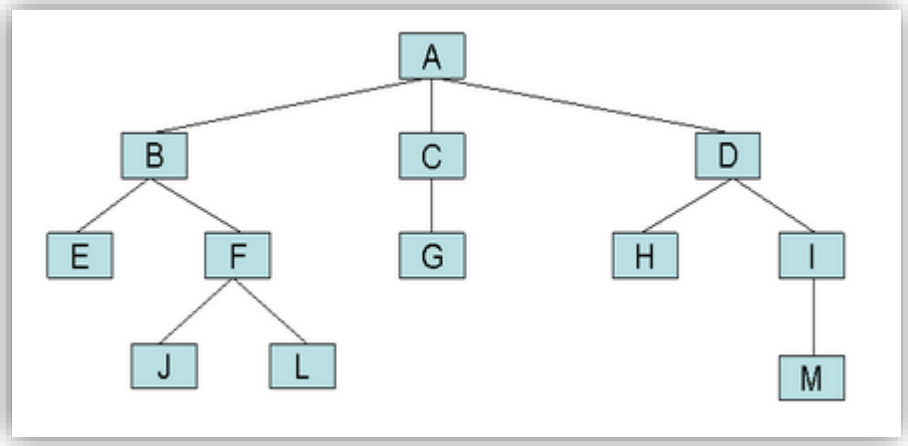
Mapa do problema





Exercícios - 1

- **Tendo com base o gráfo abaixo. Qual é o caminho que leva do estado inicial "A" para o "G" com o uso de uma política de busca em largura?**



a) Caminho = {A,B,C,G}.

b) Caminho = {A,B,E, F,C,G}.

c) Caminho = {A,C,G}.

d) Caminho = {A,B,E,F,J,L,C,G}.

e) Caminho = {A,B,C,D,E,F,G}

Exercícios - 2

- **A busca em largura é classificada como uma "busca cega", onde o conhecimento sobre o problema não é considerado. Entretanto este tipo de busca pode ser util em problemas com algumas caracteristicas. Quais caracteristicas podemos enunciar?**

a) Espaço de estados infinito e as ações tem o mesmo custo.

b) Espaço de estados finito e as ações tem custo diferente para cada transição de estados.

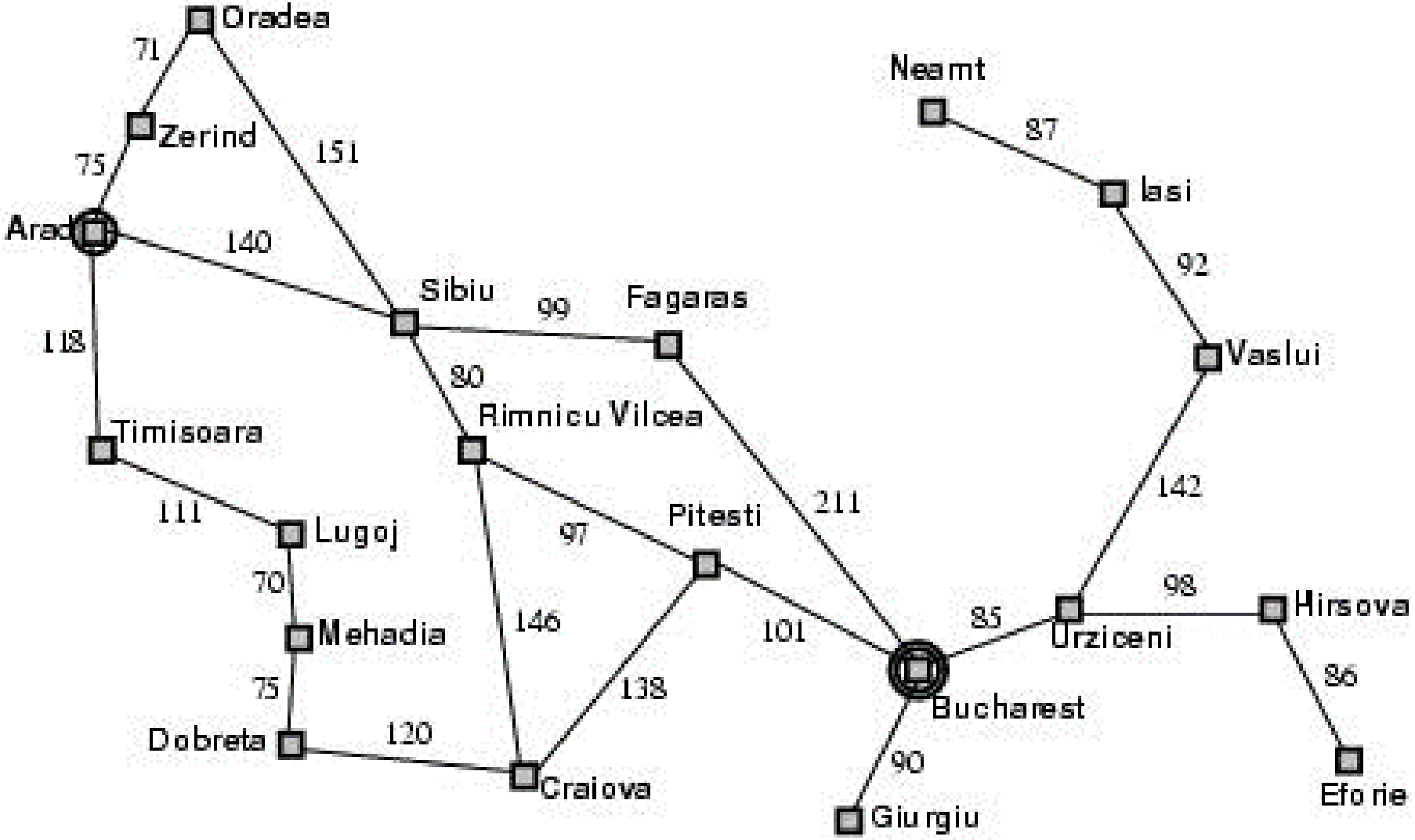
c) Espaço de estados infinito e as ações possuem custo diferenciado.

d) Espaço de estados finito e as ações tem o mesmo custo

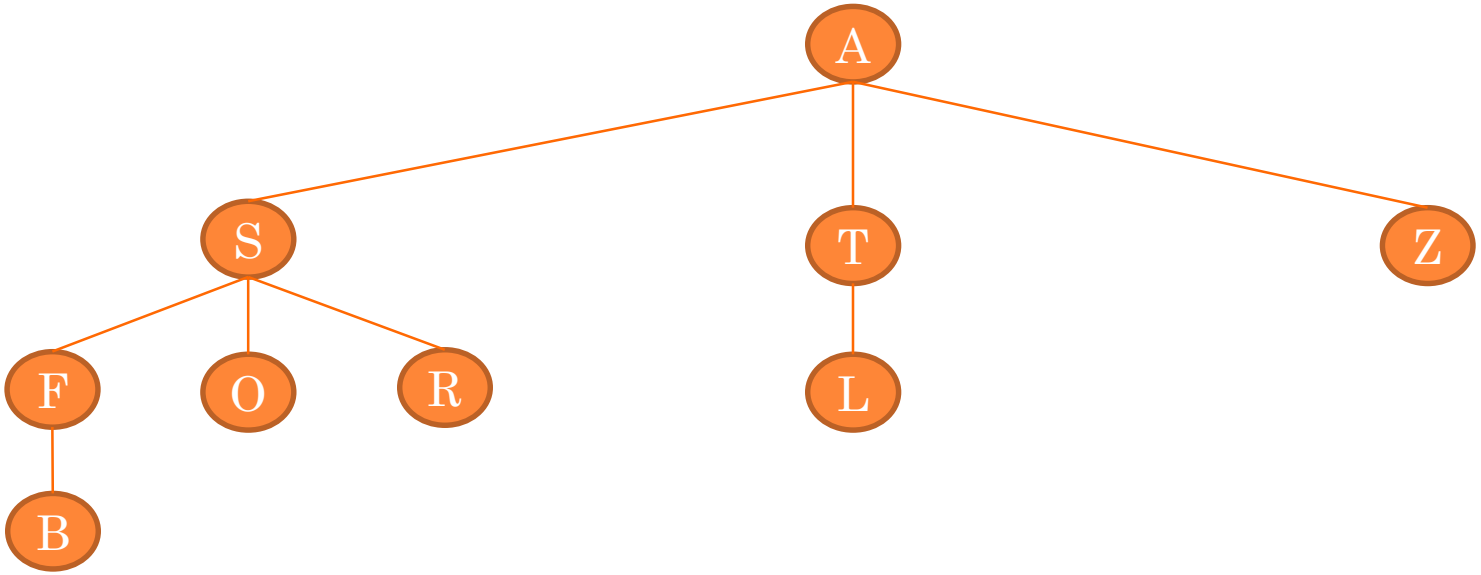
Exercícios - 3

- Faça a árvore e encontre o caminho de ARAD para BUCHAREST, utilizando o Estratégia de busca em largura.
- **Definição do problema:**
  - **Estado Inicial:** ARAD
  - **Estado Final:** BUCHAREST
  - **Ações Possíveis:** Ir(Cidade, PróximaCidade)
  - **Espaço de Estados:** Mapa da Romênia
  - **Custo**: Distância em KM entre as cidades.

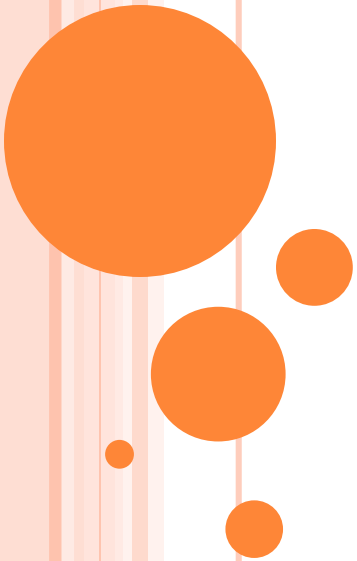
EXERCÍCIOS - 3



Solução



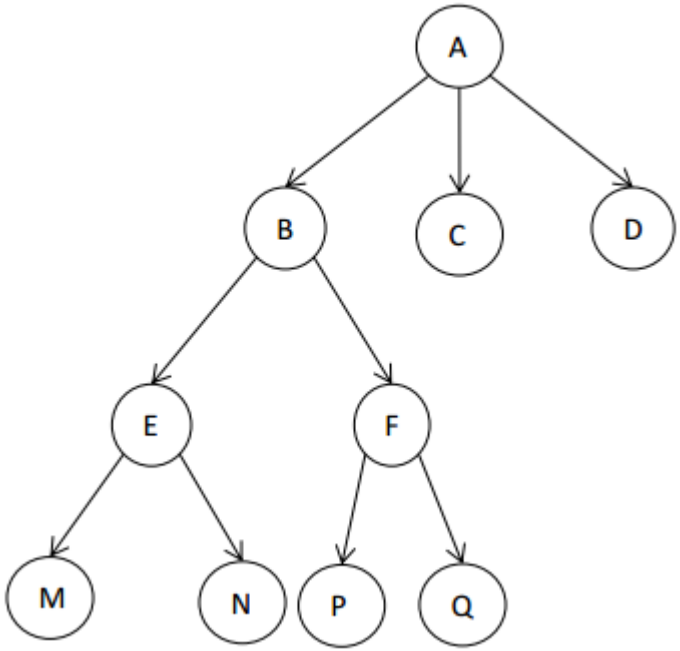
Caminho = {A,S,T,Z,F,O,R,L,B}



Busca em Profundidade (*Depth-First Search*) e Profundidade Limitada

Estratégia

- Expande os nós da vizinhança até o nó mais profundo.



Complexidade

- **Completo:** não. Caso o espaço de estados possuir uma profundidade infinita.
- **Ótimo:** não.
- **Tempo:** explorar O(bd+1) nós

  b = fator de ramos (ramificação)

  d = profundidade do estado meta

  O(bd) = b + b2 + b3 + ... + bd + (bd – b)
- Pode ser melhor do que Breadth-First Search (Busca em largura)
- **Espaço:** guardar O(bm) nós.
- Busca em profundidade requer menos espaço em memória do que a busca em largura

Busca em Profundidade

- Pode ser implementado com base no código da função BuscaEmArvore apresentado anteriormente.

- Utiliza-se uma estrutura de **pilha** *(last-in-first-out)* para armazenar os nós das fronteira.

- Pode também ser implementado de forma recursiva.

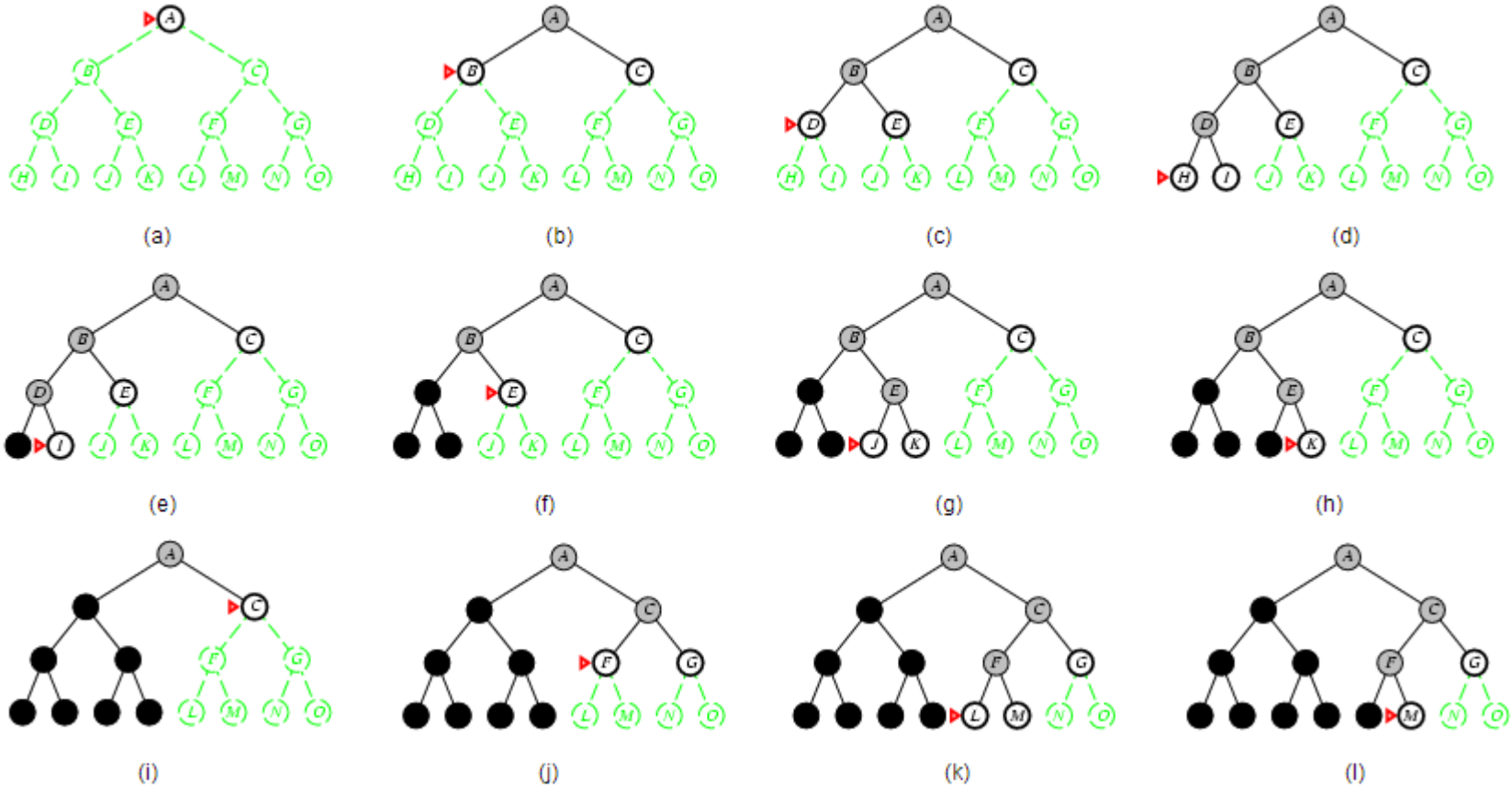
- Consome pouca memória, apenas o caminho de nós sendo analisados precisa armazenado. Caminhos que já foram explorados podem ser descartados da memória.

Busca em Profundidade

- **Vantagem:**
  - Uso de memória pela busca em largura em uma arvore com 12 de profundidade: 1000 TB.
  - Uso de memória pela busca em profundidade em uma arvore com 12 de profundidade: 118 KB.

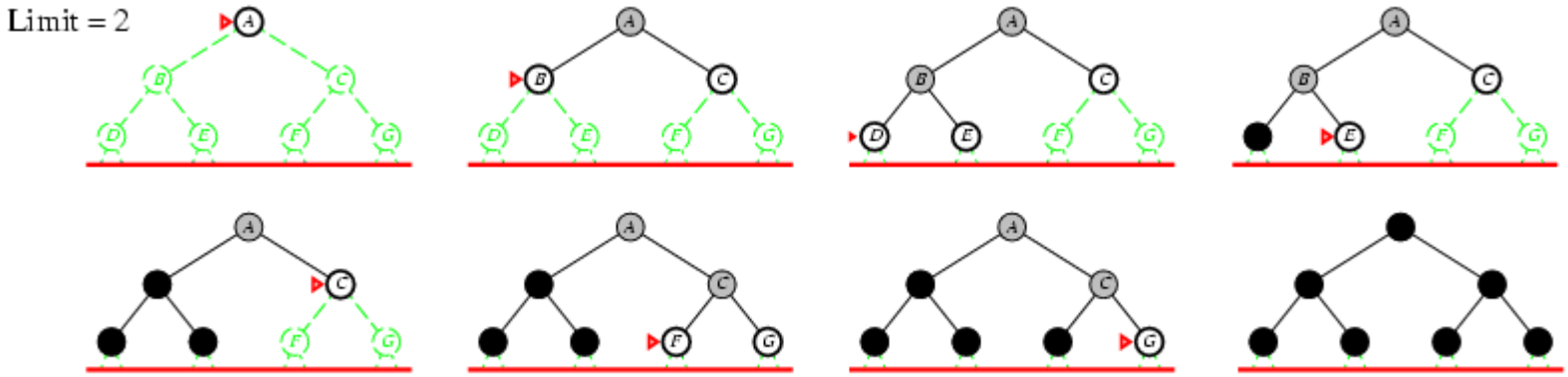
- **Problema:** O algoritmo pode fazer uma busca muito longa mesmo quando a resposta do problema esta localizado a poucos nós da raiz da árvore.

Busca em Profundidade



Depth-Limited Search (Busca em profundidade limitada)

- Representa uma Pesquisa com profundidade limitada, pré-estabelecida e fixa.



Exercícios - 1

- Quais das alternativas abaixo não se refere as características da Busca em Profundidade

a) Expande o nó no nível mais profundo da árvore.

b) A borda (fringe) é uma fila do tipo LIFO ou Pilha (Stack).

c) A borda (fringe) é uma fila do tipo FIFO

d) Para problemas de várias soluções, pode ser bem rápida.

e) Para problemas onde as árvores geradas são muito profundas ou geram caminhos infinitos, esta estratégia deve ser evitada.

Exercícios - 2

- Faça a árvore e encontre o caminho de ARAD para BUCHAREST, utilizando o Estratégia de busca em profundidade.
- **Definição do problema:**
  - **Estado Inicial:** ARAD
  - **Estado Final:** BUCHAREST
  - **Ações Possíveis:** Ir(Cidade, PróximaCidade)
  - **Espaço de Estados:** Mapa da Romênia
  - **Custo**: Distância em KM entre as cidades.

Busca em Profundidade Iterativa

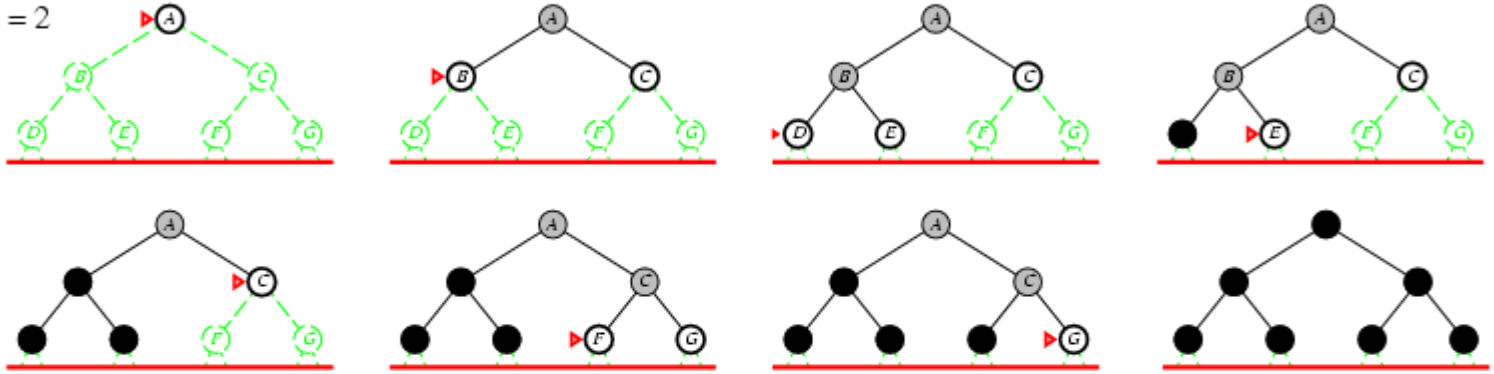
Definição

- Representa uma busca em profundidade com profundidade que se altera ao longo de cada interação (Soma + 1).
- Em geral, a busca em profundidade interativa é a pesquisa não informada preferida quando existe **um largo espaço de pesquisa e a profundidade não é conhecida**.

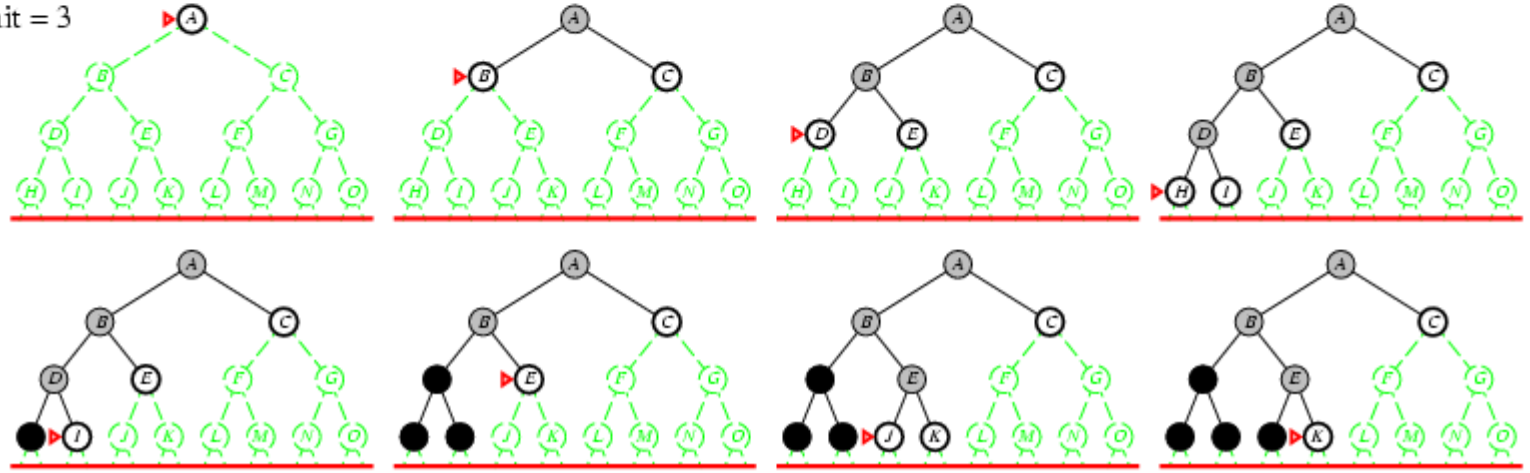
Limit = 0

Limit = 1

Limit = 2



Limit = 3



Exercícios - 1

- **Para resolver um problema cujo espaço de estados tende ao infinito, qual das estratégias de buscas cegas seriam mais apropriadas, caso o nó objetivo encontra-se em profunidade próxima à superfície do espaço de estados?**

a) Busca em profundidade.

b) Busca em profundidade iterativa

c) Busca em largura.

d) Busca em em profundidade iterativa acrescentando apenas uma iteração.

Exercícios - 2

- O processo de expansão em uma árvore de busca consiste em:

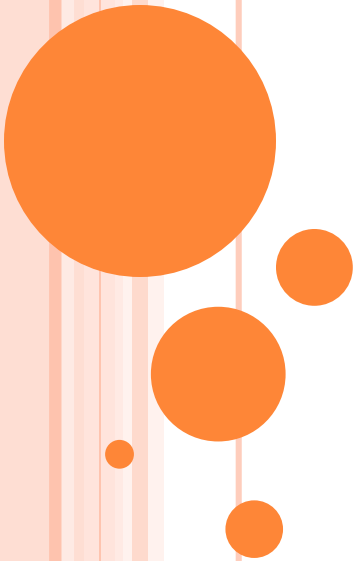
a) Ocorre após a verificação do NÓ e consiste na geração do ESPAÇO DE ESTADOS que são uma seqüência de nós proveniente do NÓ-PAI. Cada nó deste ESPAÇO DE ESTADOS é representado por cinco elementos que o caracterizam.

b) Ocorre antes da verificação NÓ e consiste na geração de um NÓ-FOLHA a partir do NÓ verificado e assim por diante até chegar na profundidade máxima da ÁRVORE-DE-BUSCA.

c) Ocorre após a verificação do NÓ e consiste na geração da borda que é uma seqüência de nós proveniente do NÓ-PAI. Cada nó desta borda é conhecido por NÓ-FOLHA

d) Ocorre durante a verificação do NÓ e consiste na execução do algoritmo do AGENTE-DE-RESOLUÇÃO-DE-PROBLEMAS, o qual resulta em uma seqüência de ações.

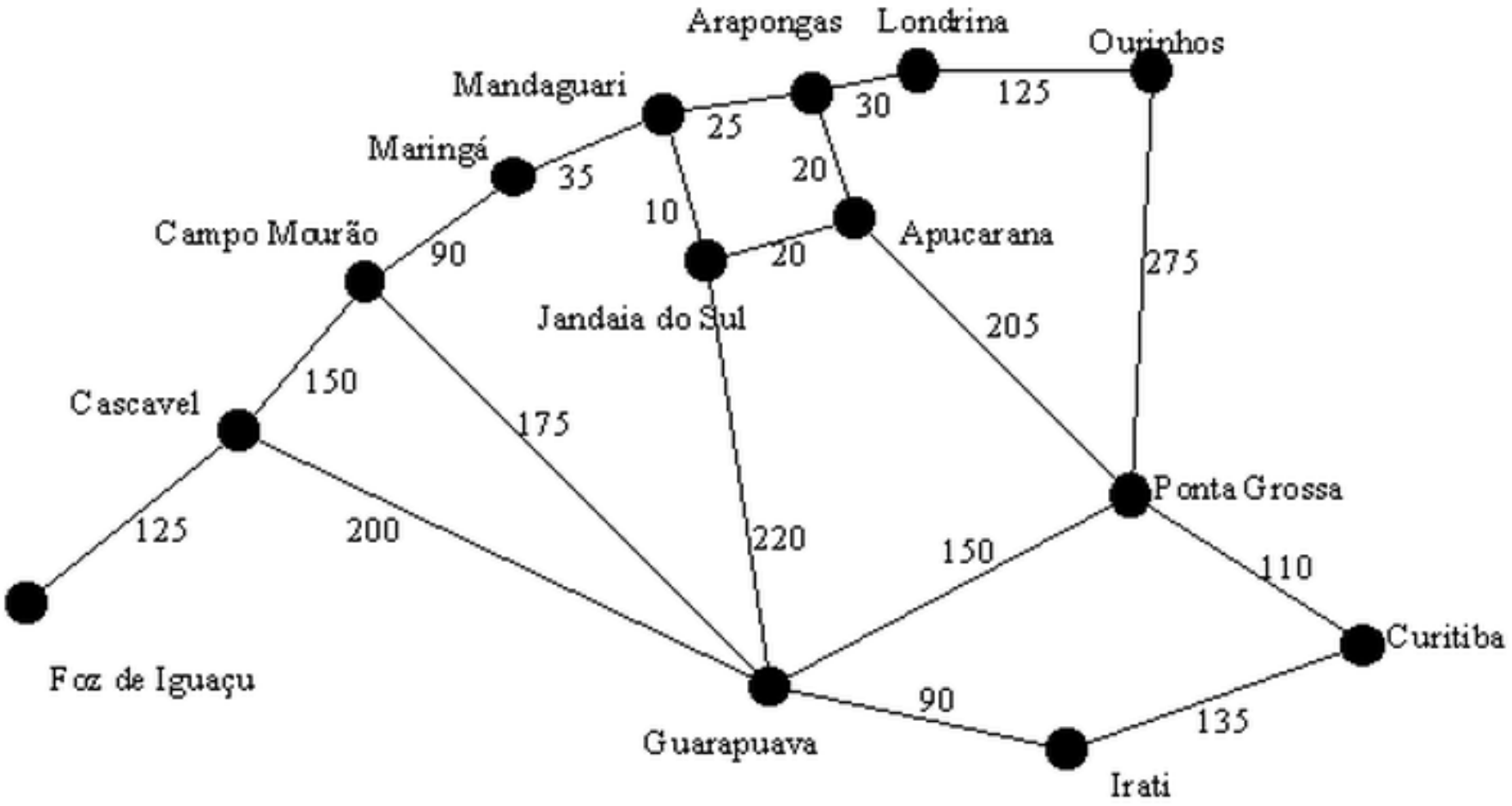
e) Ocorre após a verificação do NÓ e consiste na geração de uma seqüência de NÓS dispostos em uma FILA do tipo PILHA, independente de qualquer estratégia de busca pré-estabelecida no algoritmo.



Exercícios

Buscas Cegas

Mapa das cidades do Paraná



Mapa das cidades do Paraná

Estado Inicial: Curitiba

Estado Final: Londrina

Ações Possíveis: Ir(Cidade,Vizinha)

Espaço de Estados: Mapa do Paraná

Custo: quilometragem.

- Busca em Largura
- Busca em Profundidade
- Busca em Profundidade Limitada
- Busca em Profundidade Iterativa

QUEBRA-CABEÇA

Estado Inicial:

1	**2**	**3**
8	**6**	**4**
	7	**5**

Estado Final:

1	**2**	**3**
8		**4**
7	**6**	**5**

Ações Possíveis: mover vazio, cima abaixo, lado esquerdo, lado direito

Espaço de Estados: Conjunto de possibilidades devido a locomoção do vazio

Custo: 1 movimento

Como evitar estados repetidos

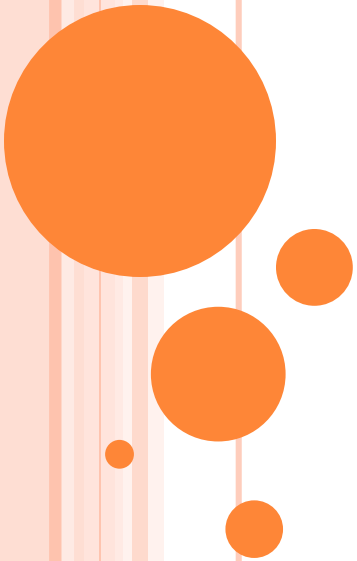
- Estados repetidos sempre vão ocorrer em problema onde os estados são reversíveis.

- Como evitar?
  - Não retornar ao estado "pai".
  - Não retorna a um ancestral.
  - Não gerar qualquer estado que já tenha sido criado antes (em qualquer ramo).
    - Requer que todos os estados gerados permaneçam na memória.

Análise dos algoritmos de busca cega

Definições

- **Completeza:** encontra uma solução quando ela existe
- **Otimização:** o menor custo de caminho entre todas as soluções
- **Complexidade de tempo:** Quanto tempo ele leva para encontrar uma solução?
- **Complexidade de espaço:** Quanta memória é necessária para executar a busca?



Busca Heurística

Busca Heurística

- A busca heurística leva em conta o **objetivo** para decidir qual caminho escolher.

- Conhecimento extra sobre o problema é utilizado para **guiar o processo de busca.**

Busca Heurística

Como encontrar um barco perdido?

- Busca Cega: Procura no oceano inteiro.

- Busca Heurística: Procura utilizando informações relativas ao problema.

  Exemplo: correntes marítimas, vento, etc.

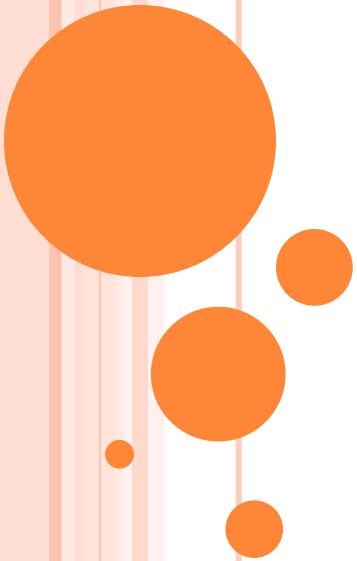
Busca Heurística

- **Função Heurística (h)**
  - Estima o custo do caminho mais barato do estado atual até o estado final mais próximo.
  - São específicas para cada problema.

Exemplo:

Encontrar a rota mais curta entre duas cidades:

h(n) = distância em linha reta direta entre o nó n e o nó final.



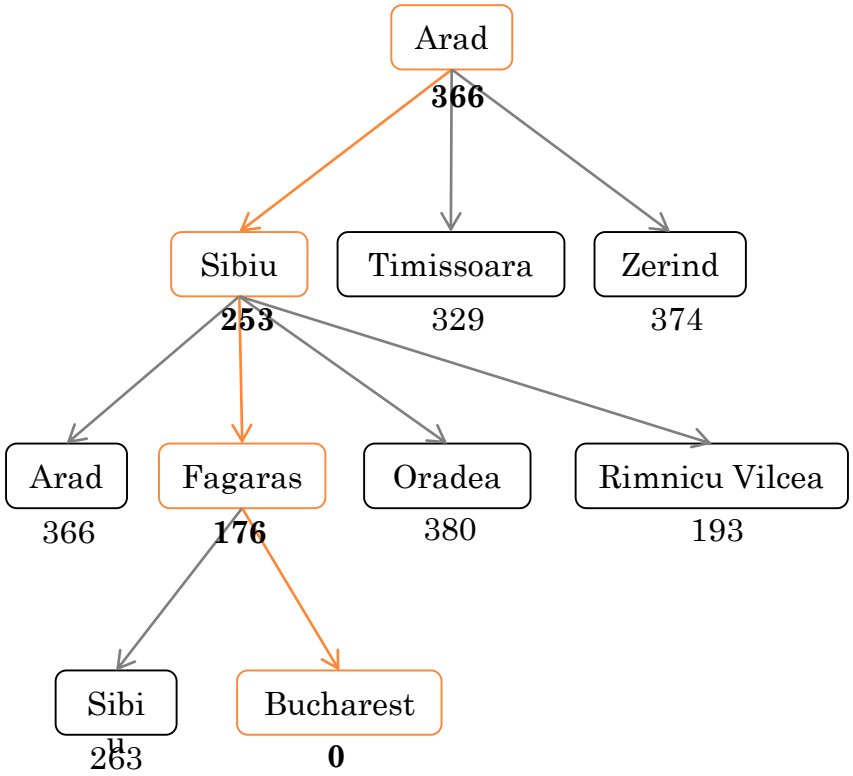
Busca Gulosa

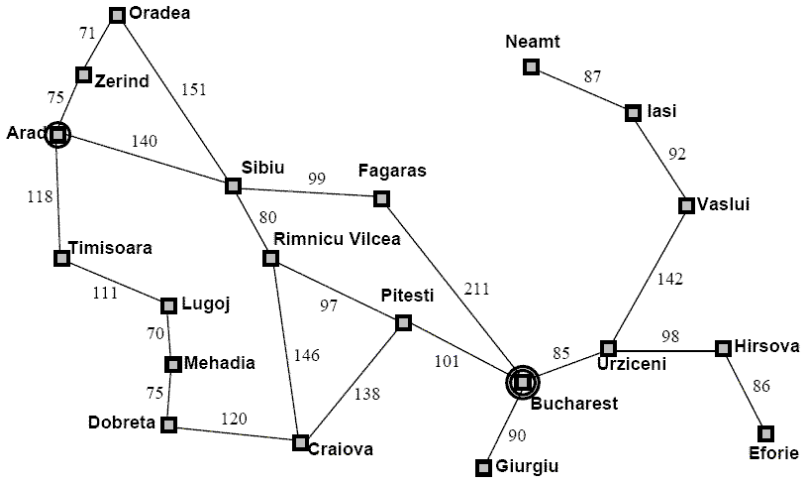
Estratégia

- Expande os nós que se encontram mais próximos do objetivo (uma linha reta conectando os dois pontos no caso de distâncias), desta maneira é provável que a busca encontre uma solução rapidamente.

- A implementação do algoritmo se assemelha ao utilizado na busca cega, entretanto utiliza-se uma função heurística para decidir qual o nó deve ser expandido.

Busca Gulosa





Função Heurística (h):
Distância em linha reta

Arad	366	Mehadia	241
Bucharest	0	Neamt	234
Craiova	160	Oradea	380
Drobeta	242	Pitesti	100
Eforie	161	Rimnicu Vilcea	193
Fagaras	176	Sibiu	253
Giurgiu	77	Timisoara	329
Iasi	226	Vaslui	199
Lugoj	244	Zerind	374
Hirsova	151	Urziceni	80

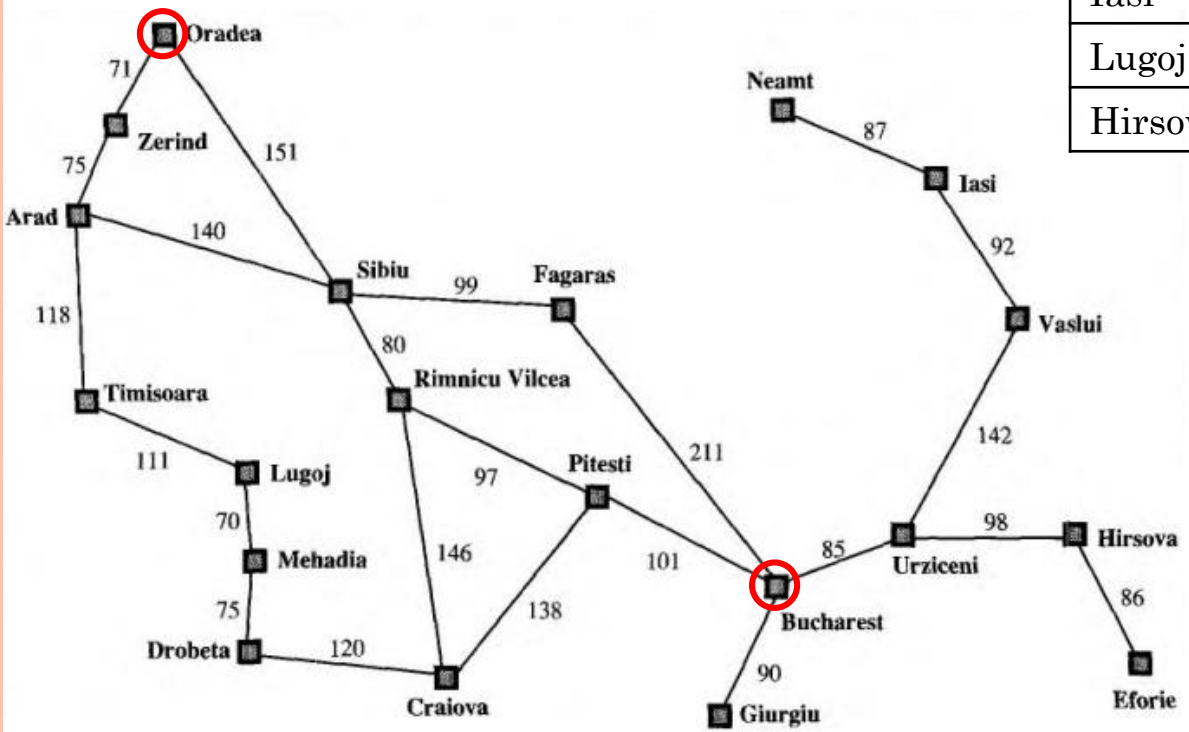
Busca Gulosa

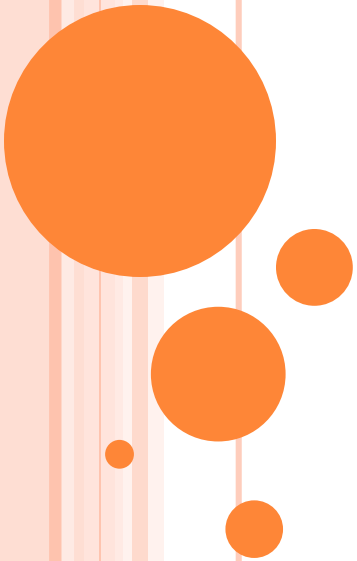
- **Custo de busca mínimo:**
  - No exemplo, não expande nós fora do caminho.
- **Não é ótima:**
  - No exemplo, escolhe o caminho que é mais econômico à primeira vista, via Fagaras (450km).
  - Porém, existe um caminho mais curto via Rimnicu Vilcea (418 km).
- **Não é completa:**
  - Pode entrar em loop se não detectar a expansão de estados repetidos.
  - Pode tentar desenvolver um caminho infinito.

Exercício

- Ir de Oradea para Bucharest?

Arad	366	Mehadia	241
Bucharest	0	Neamt	234
Craiova	160	Oradea	380
Drobeta	242	Pitesti	100
Eforie	161	Rimnicu Vilcea	193
Fagaras	176	Sibiu	253
Giurgiu	77	Timisoara	329
Iasi	226	Vaslui	199
Lugoj	244	Zerind	374
Hirsova	151	Urziceni	80



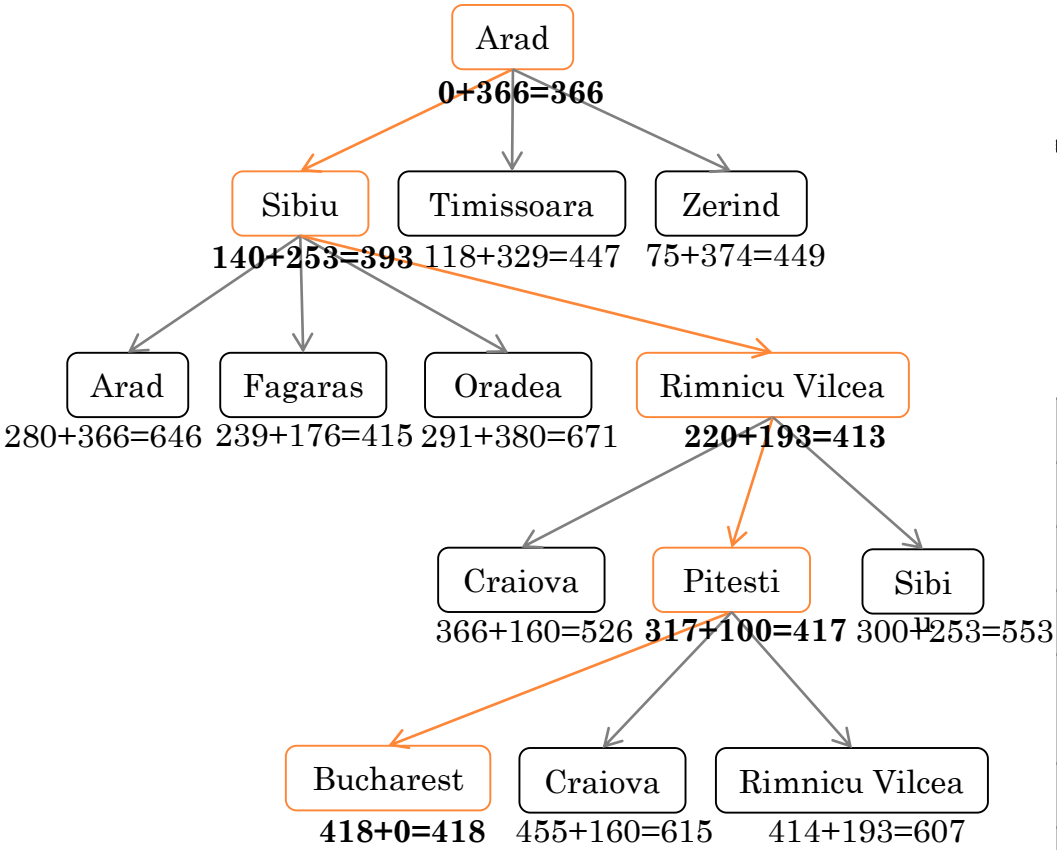


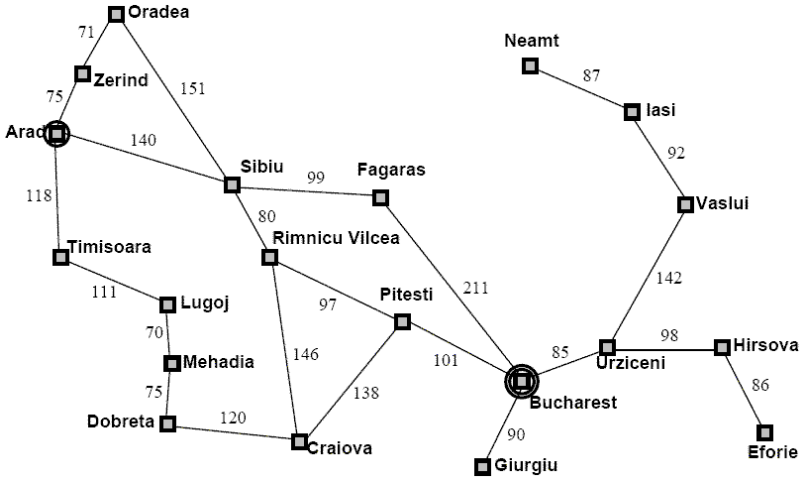
Busca A*

Estratégia

- Combina o custo do caminho g(n) com o valor da heurística h(n);
- g(n) = custo do caminho do nó inicial até o nó n
- h(n) = valor da heurística do nó n até um nó objetivo (distância em linha reta no caso de distâncias espaciais)
- f(n) = g(n) + h(n)
- Sempre seleciona o nó da fronteira com menor f(n);
- É a técnica de busca mais utilizada.

Busca A*





Função Heurística

h(n): Distância em linha reta

Arad	366	Mehadia	241
Bucharest	0	Neamt	234
Craiova	160	Oradea	380
Drobeta	242	Pitesti	100
Eforie	161	Rimnicu Vilcea	193
Fagaras	176	Sibiu	253
Giurgiu	77	Timisoara	329
Iasi	226	Vaslui	199
Lugoj	244	Zerind	374
Hirsova	151	Urziceni	80

Busca A*

- A estratégia é **completa**, pois sempre chega em uma solução e **ótima** porque não existe outra solução com melhor desempenho de f(n).

- **Custo de tempo:**
  - Exponencial com o comprimento da solução, porém boas funções heurísticas diminuem significativamente esse custo.

- **Custo memória:** $O(b^d)$
  - Guarda todos os nós expandidos na memória.

- Nenhum outro algoritmo ótimo garante expandir menos nós.

Heurística Admissível

- Se existe uma solução, A* sempre encontra a solução ótima se:
  - O fator de ramificação for finito;
  - O custo do caminho nunca decresce (os arcos não possuem custos negativos);
  - h(n) nunca superestima o custo do caminho;
  - h(n) <= o custo real do caminho (heurística admissível);

Exercícios

- **Conceitue heurística referente a busca no espaço de estados.**

a) Heurística representa um busca.

b) Heurística é um modelo matemático idealizado para criar um caminho com atalhos no espaço de estados

c) Heurística é um modelo de raciocínio lógico, pouco utilizado pelas buscas cegas.

d) Heurística é o caminho mais longo entre o nó inicial e objetivo.

Exercícios

- Em um dado problema necessitamos resolvê-lo com o uso de heurística. Quantas heurísticas podemos ter para resolver um único problema?

a) Apenas 1 heurística .
b) Quantas heurísticas o problema necessitar
c) Nenhuma heurísticas .
d) Somente 2 heurísticas .

Exercícios

- Uma das formas de avaliação das buscas em inteligência artificial é através de critérios bem estabelecidos. Quais são os critérios de avaliação no que diz respeito a complexidade de uma estratégia de busca?

a) Tempo e Espaço.
b) Completo, Ótimo, Complexidade de Tempo e Espaço
c) Planejamento, Análise e Pontos por função.
d) Indução e Dedução do conhecimento.

Exercícios

- **A busca A* é uma pesquisa em árvore que considera informações globais e locais. A busca A* somente é completa se:**

a) A heurística é persistente.

b) A heurística é global.

c) A heurística é local.

d) A heurística é consistente

Exercícios

- **O algoritmo A* pode ser implementado através de uma estrutura de dados do tipo árvore. Escolha a alternativa correta que demonstre esta estrutura de dados.**

a) struct node {int item; struct node *prox };

b) struct node{ int item; struct node *esq, *dir };

c) struct node{ int item; struct node *esq};

d) struct node{ int item; struct node *dir };

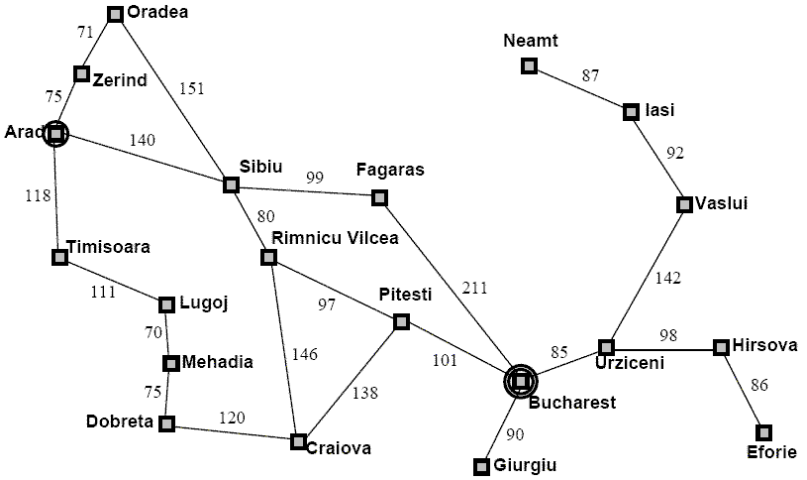
Busca Recursiva Best-first (RBFS)

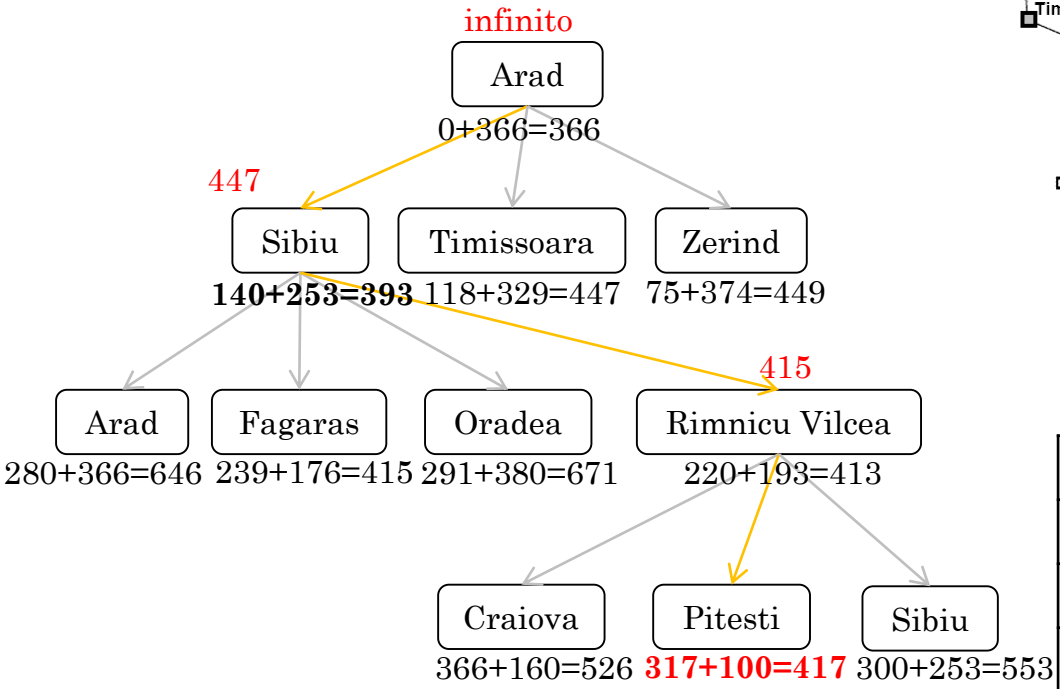
Busca Recursiva Best-first (RBFS)

- É um algoritmo recursivo que tenta imitar a operação da busca pelo melhor (Best Fit), mas utilizando apenas o espaço linear.
- Busca:
  - O nó do estado inicial é calculado o f(n) e o seu limite é configurado como infinito;
  - Expande-se o nó inicial;
  - a) Para cada filho calcula-se f(n) o melhor valor recebe como limite o próximo valor mais baixo entre os filhos e o pai;
  - b) Verifica-se se o nó objetivo está entre os filhos e se possui um valor menor que o limite do nó ancestral e mostra o resultado (termina algoritmo)
  - c) Caso não haja mais sucessores retorna falha;
  - d) Caso ainda haja sucessores e o nó objetivo ainda não foi encontrado, expandir os sucessores e calcular f(n), selecionar o nó com o melhor valor, caso ele possua f(n) maior que o seu ancestral o ancestral recebe f(n) o valor deste nó e o restante da árvore é esquecida. Repete-se b,c,d.

RBFS

Limite:
O menor valor entre os sucessores que não foram explorados





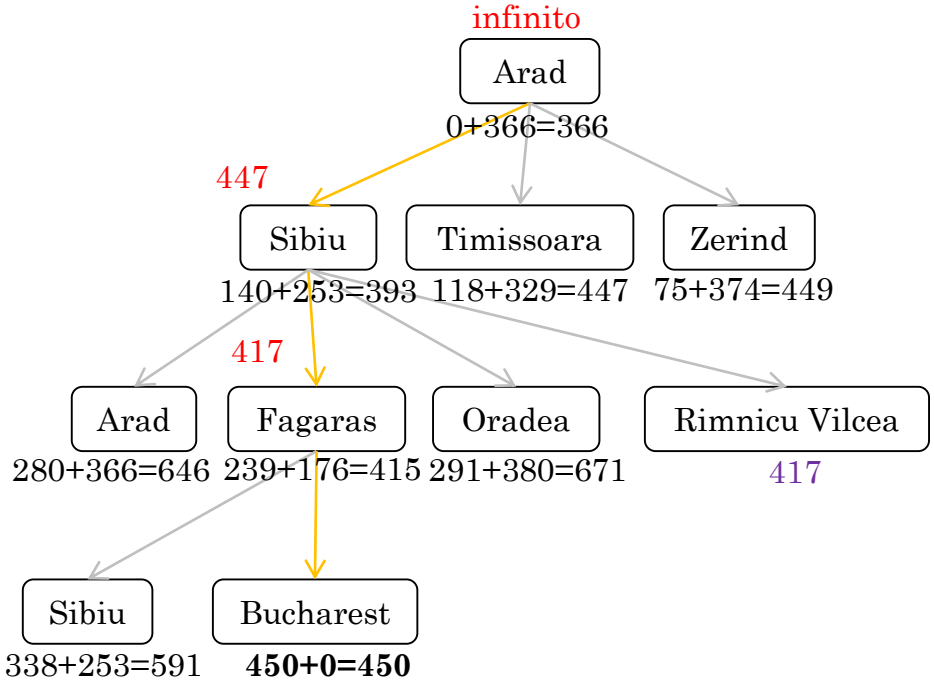
A expansão ocorre fazendo o cálculo de f(n) = g(n) + h(n) cada nó.
Dentre os sucessores expandidos escolhe-se o de menor valor para seguir a expansão.
O segundo menor valor entre todos os nós não visitados é selecionado como limite da expansão corrente. Até que estoure o limite como e o caso de Pitesti (417)

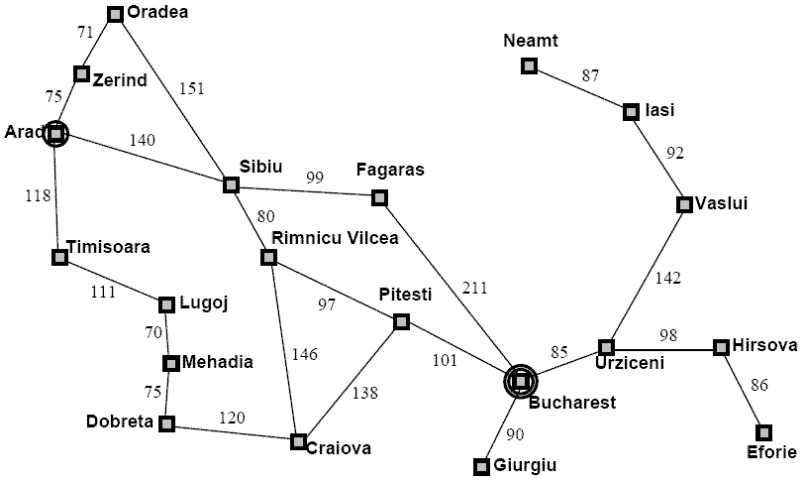
Função Heurística
h(n): Distância em linha reta

Arad	366	Mehadia	241
Bucharest	0	Neamt	234
Craiova	160	Oradea	380
Drobeta	242	Pitesti	100
Eforie	161	Rimnicu Vilcea	193
Fagaras	176	Sibiu	253
Giurgiu	77	Timisoara	329
Iasi	226	Vaslui	199
Lugoj	244	Zerind	374
Hirsova	151	Urziceni	80

RBFS

Limite:
O menor valor entre os sucessores que não foram explorados





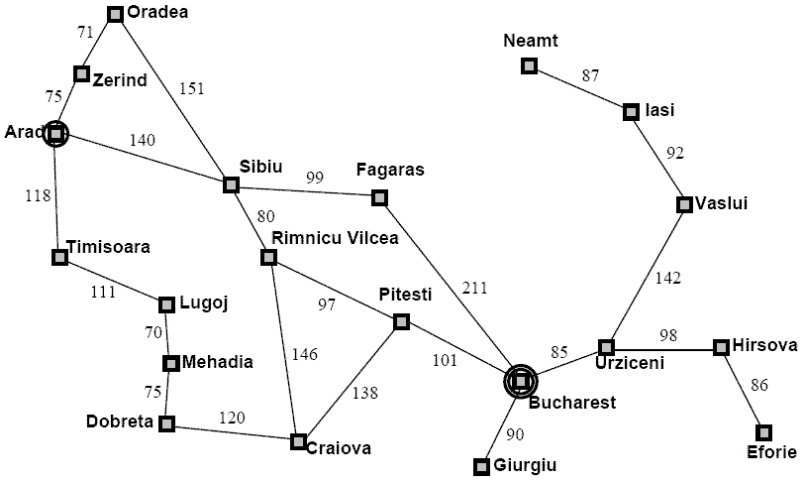
O pai que gerou o nó que passou do limite recebe como novo valor, o valor do estouro Pitesti (417).
O novo limite do nível passa a ser o segundo menor valor de todos os nós não visitados (agora 417). Então o novo limite da camada passa a ser 417 e não 415.
Então o menor valor passa a ser Fagaras (415) que é expandido até estourar o limite Bucharest (450).
O no pai que gerou o estouro recebe o valor do estouro (450).

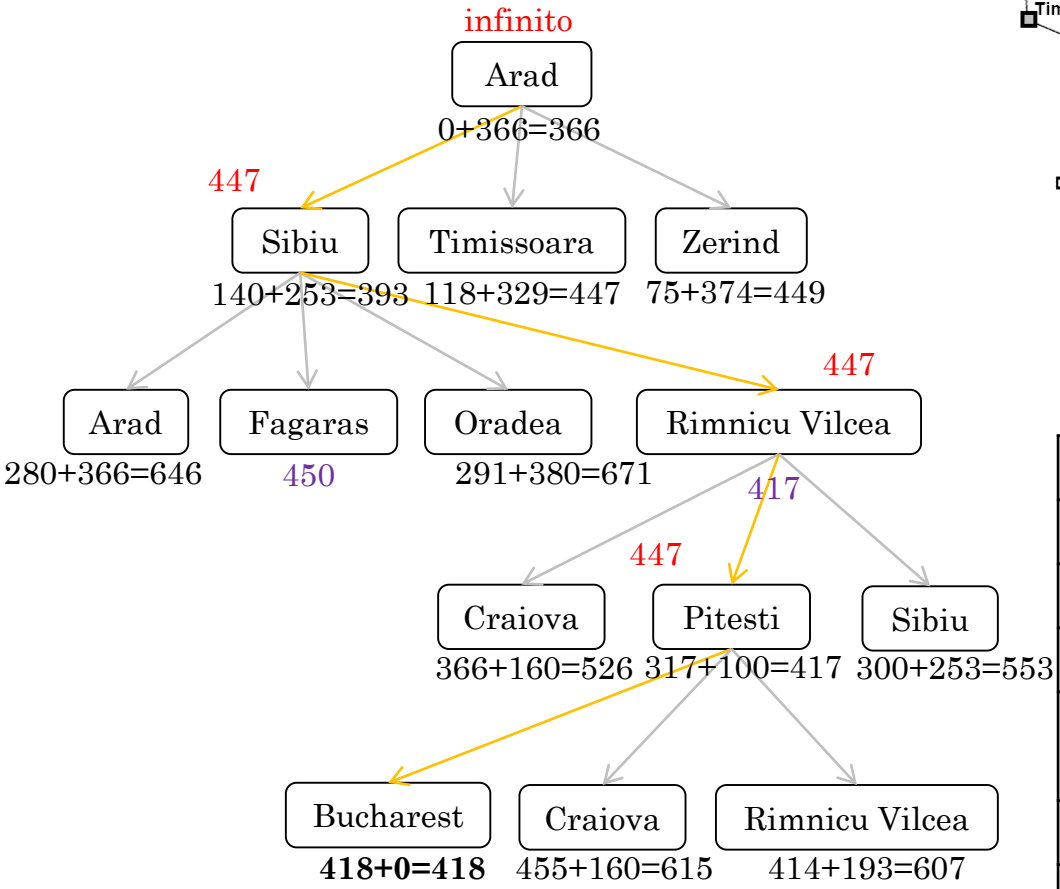
Função Heurística
h(n): Distância em linha reta

Arad	366	Mehadia	241
Bucharest	0	Neamt	234
Craiova	160	Oradea	380
Drobeta	242	Pitesti	100
Eforie	161	Rimnicu Vilcea	193
Fagaras	176	Sibiu	253
Giurgiu	77	Timisoara	329
Iasi	226	Vaslui	199
Lugoj	244	Zerind	374
Hirsova	151	Urziceni	80

RBFS

Limite:
O menor valor entre os sucessores que não foram explorados



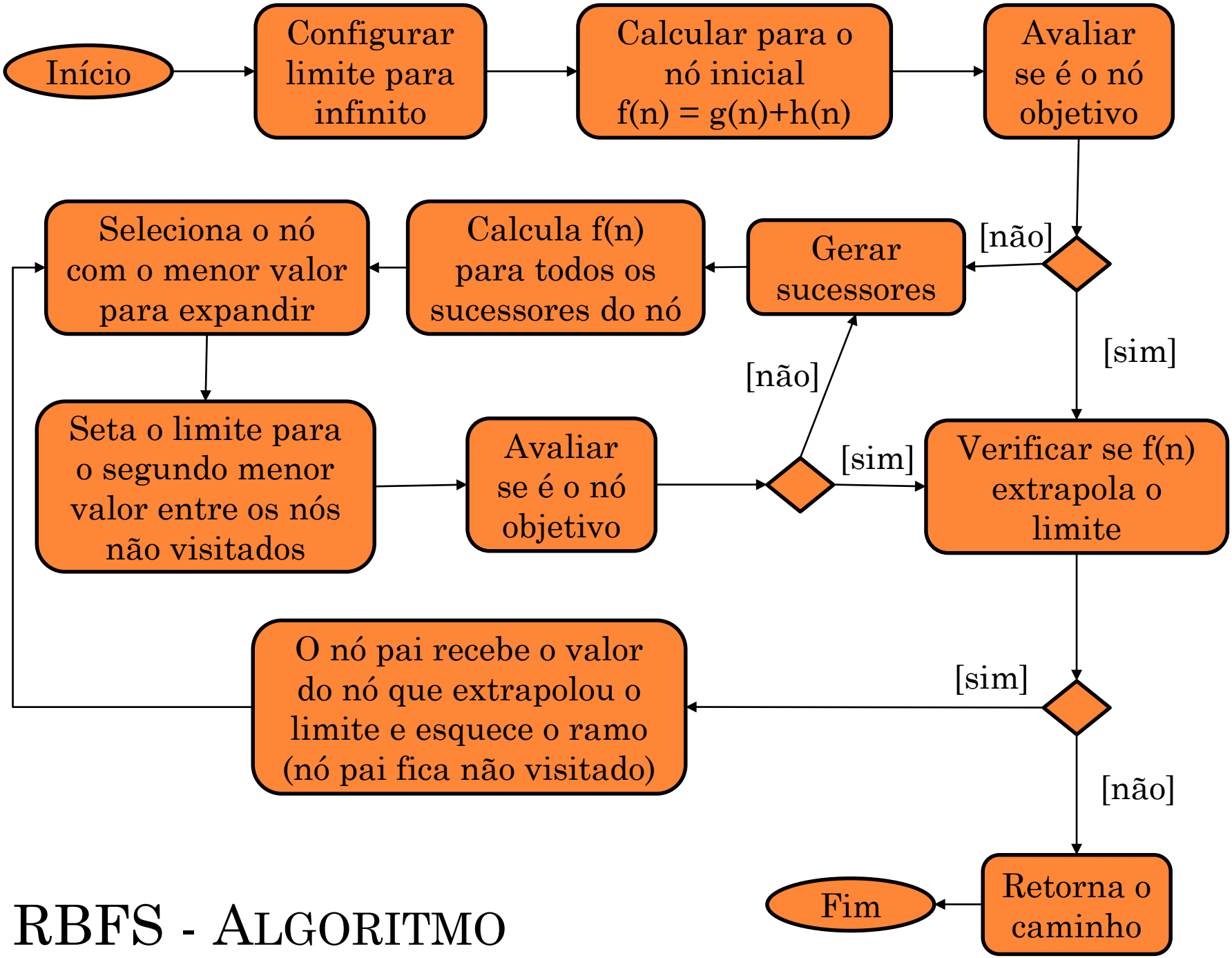


Como Fagaras agora é 450 o segundo menor valor de todos os nós não visitados agora é 447 (Timissoara)
Portanto o menor valor é Rimnicu que é expandido. O menor valor dos seus sucessores é Piteste e o segundo menor valor de todos os nós não visitados para esse nível é 447, portanto o limite dessa camada fica 447. E então expande Pitesti.

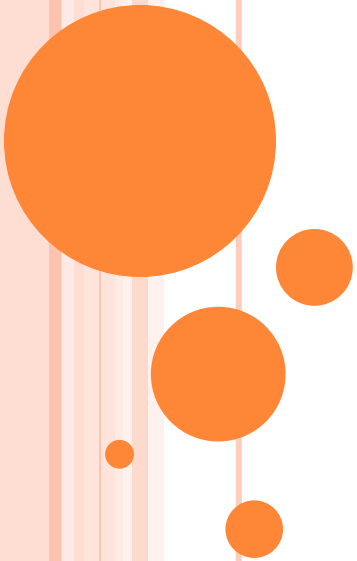
Função Heurística

h(n): Distância em linha reta

Arad	366	Mehadia	241
Bucharest	0	Neamt	234
Craiova	160	Oradea	380
Drobeta	242	Pitesti	100
Eforie	161	Rimnicu Vilcea	193
Fagaras	176	Sibiu	253
Giurgiu	77	Timisoara	329
Iasi	226	Vaslui	199
Lugoj	244	Zerind	374
Hirsova	151	Urziceni	80



RBFS - Algoritmo



Busca Escalada na Montanha (*Hill-Climbing*) e Algoritmos Genéticos

Busca Local

- Em muitos problemas o **caminho para a solução é irrelevante**.
  - **Jogo das n-rainhas:** o que importa é a configuração final e não a ordem em que as rainhas foram acrescentadas.
  - **Outros exemplos:**
    - Projeto de Circuitos eletrônicos;
    - Layout de instalações industriais;
    - Escalonamento de salas de aula;
    - Otimização de redes;
- Se o caminho para a solução não importa, podemos utilizar um algoritmo de **busca local**.

Busca Local

- Algoritmos de busca local operam sobre um **único estado corrente**, ao invés de vários caminhos.

- Em geral se movem apenas para os **vizinhos** desse estado.

- O caminho seguido pelo algoritmo não é guardado.

Busca local

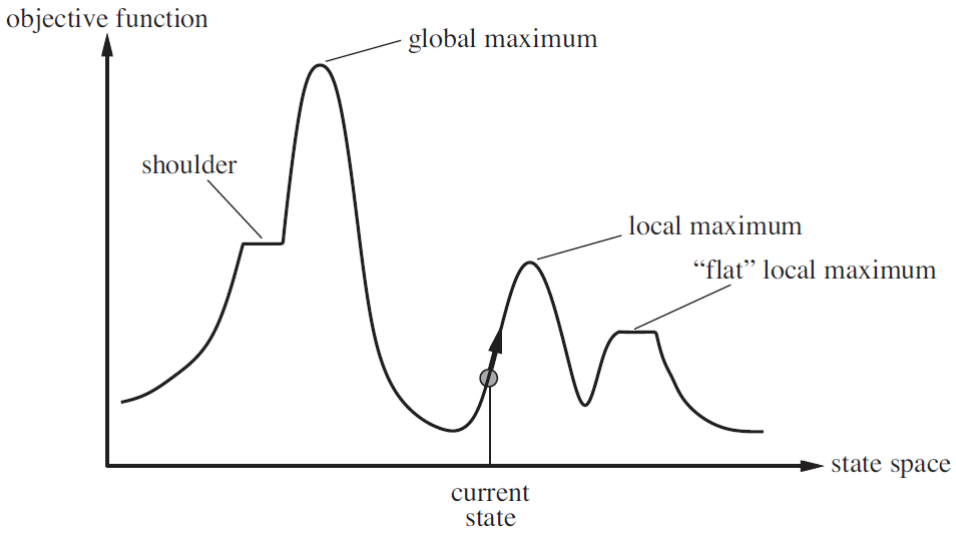
- **Vantagens:**
  - Ocupam pouquíssima memória (normalmente constante).
  - Podem encontrar soluções razoáveis em grandes ou infinitos espaços de estados.

- **São uteis para resolver problemas de otimização.**
  - Buscar por estados que atendam a uma função objetivo.

Busca local

- **Panorama do Espaço de Estados**

- **Local** = Estado;
- **Elevação** = Valor de custo da função heurística;
- Busca-se o máximo ou mínimo global;



Busca Escalada na Montanha (*Hill-Climbing*)

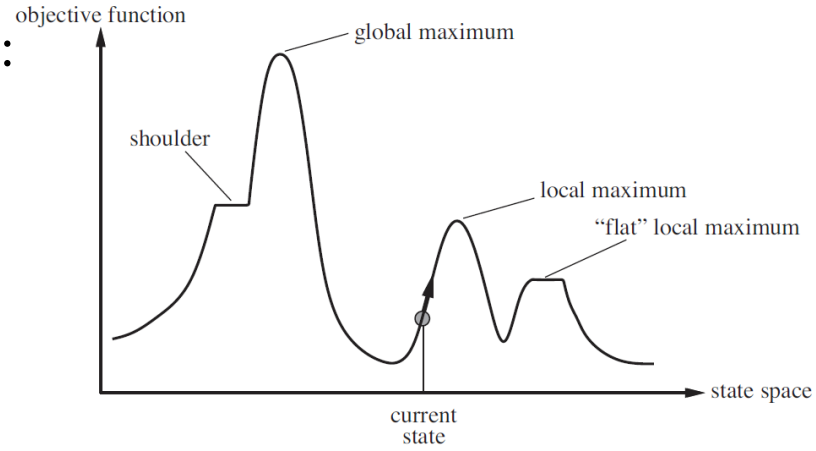
- Consiste de de um loop que continuamente move-se para os estados que aumentam o valor em sua função de avaliação.
- Termina quando atinge um "pico" onde nenhum vizinho tem um valor maior.
- Não mantem uma árvores de busca.

Busca Escalada na Montanha (*Hill-Climbing*)

- É um **algoritmo guloso** – escolhe sempre o primeiro melhor vizinho para progredir na busca.

- Essa abordagem pode ter **bons resultados em alguns problemas**. Sendo capaz de progredir rapidamente para a solução problema.

- Mas, sofre de três sérios **problemas**:
  - Máximos locais
  - Planícies
  - Encostas e Picos



Busca Escalada na Montanha (*Hill-Climbing*)

- **Variações:**
  - *Random-Restart Hill Climbing;*

- **Não é ótimo e não é completo.**

- O desempenho do *Hill Climbing* depende muito do **formato do panorama** do espaço de estados.

Algoritmos Genéticos

- Método de **busca local**.

- Boa abordagem para lidar com **espaços de busca muito grandes**.
  - Possibilita navegá-los procurando por soluções que talvez não fossem encontradas em uma busca convencional mesmo que ela durasse centenas de anos.

- Baseado na **evolução dos seres biológicos.**

Teoria da Evolução

- A **teoria da evolução** diz que na natureza todos os indivíduos dentro de um ecossistema **competem** entre si por recursos limitados (comida, água...)

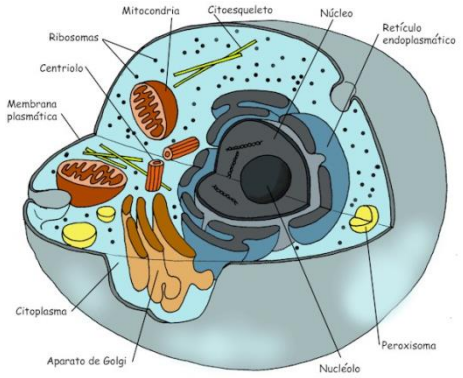
- Os **indivíduos mais fracos** de uma mesma espécie tendem a não se proliferarem.

- A descendência reduzida faz com que a probabilidade de ter seus **genes propagados** ao longo de sucessivas gerações seja menor.

- A combinação entre os genes dos indivíduos que sobrevivem pode produzir um novo **indivíduo muito melhor adaptado** às características de seu meio ambiente ao combinar características possivelmente positivas de cada um dos seus pais.

Relembrando as Aulas de Biologia

- Todo indivíduo biológico é formado por uma ou mais **células**.

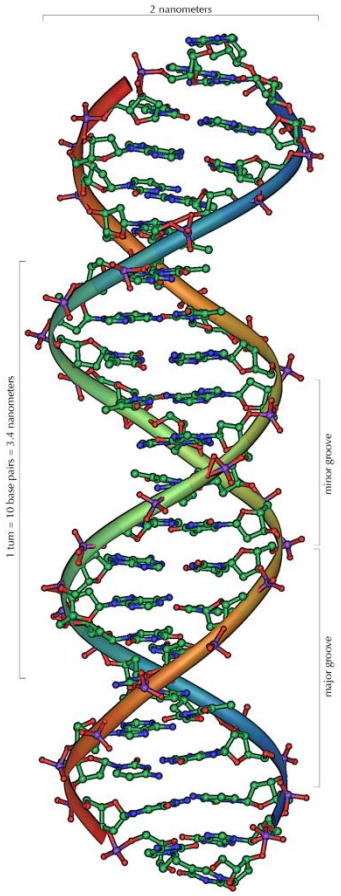


- Dentro de cada célula existe um conjunto de **cromossomos**.

- Os seres humanos têm 23 pares de cromossomos por célula.
  - O número de pares varia de espécie para espécie.

Relembrando as Aulas de Biologia

- Um cromossomo consiste em sequências de **DNA**.
  - DNA = molécula que codifica toda a informação necessária para o desenvolvimento e funcionamento de organismos vivos.
- Um cromossomo possui vários **genes** (blocos de sequências de DNA).
- Cada gene tem uma posição própria no cromossomo.
- O conjunto completo de material genético (todos os cromossomos) é chamado de **genoma**.
- A **qualidade de um indivíduo** (fitness) é medida pelo seu sucesso (sobrevivência)



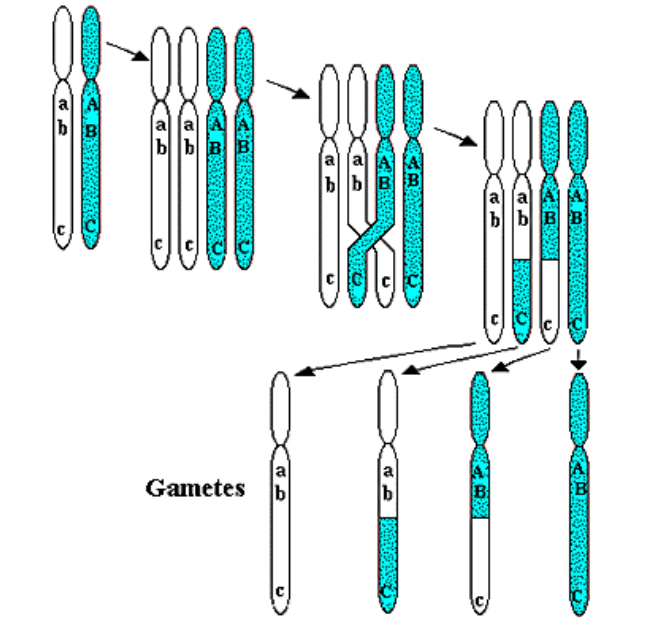
Relembrando as Aulas de Biologia

- Na natureza existem dois tipos de **reprodução**:
  - **Assexuada**: típica de organismos inferiores, como bactérias.
  - **Sexuada**: exige a presença de dois organismos, na maioria das vezes de sexos opostos, que trocam material genético.
- **Reprodução assexuada** é base para o algoritmo de busca local **Beam Search**.
- **Reprodução sexuada** é a base dos **algoritmos genéticos**.

Relembrando as Aulas de Biologia

- Na **reprodução sexuada** ocorre a formação de um **novo indivíduo** através da combinação de duas células gametas.

- Na formação destas gametas, ocorre o processo de **recombinação genética** (crossing-over).



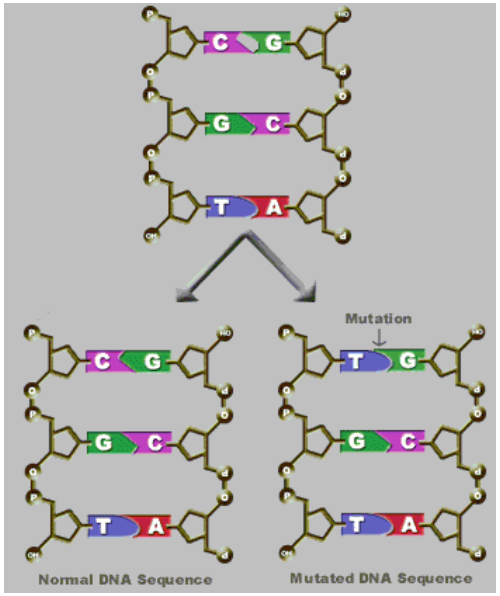
Crossing-over and recombination during meiosis

Relembrando as Aulas de Biologia

- O processo de replicação do DNA é extremamente complexo.

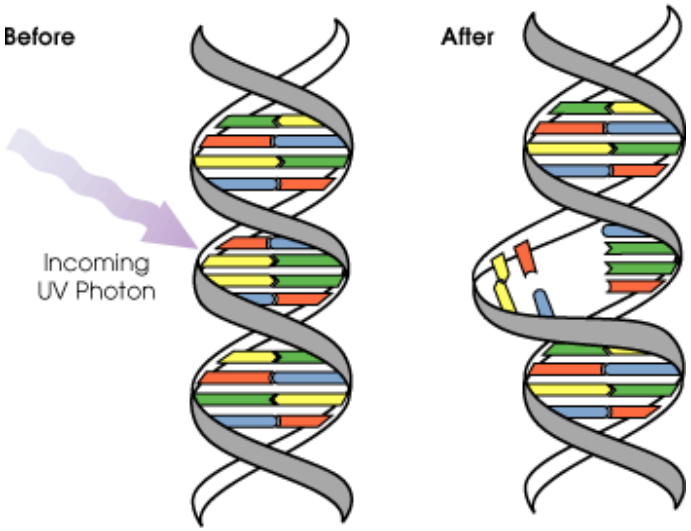
- Pequenos **erros** podem ocorrer ao longo do tempo, gerando **mutações** dentro do código genético.

- Estas mutações podem ser boas, ruins ou neutras.



Relembrando as Aulas de Biologia

- Alguns **fatores externos**, como a radiação ultravioleta, também podem causar pequenas disrrupções no código genético.



Relembrando as Aulas de Biologia

- Indivíduos com uma melhor adequação do seu fenótipo ao meio ambiente (**melhor fitness**) se reproduzem mais.

- Dessa forma têm mais chances de passar seus genes para a **próxima geração**.

- Entretanto, graças aos **operadores genéticos** (recombinação e mutação) os cromossomos dos filhos não são exatamente iguais aos dos pais.

- Assim, eles podem **evoluir** e se **adaptar** cada vez mais aos meio ambiente que os cerca.

Algoritmos Evolucionários

- Os **algoritmos evolucionários**, dos quais os algoritmos genéticos fazem parte, procuram se inspirar na forma como a natureza funciona.

  - Algoritmos Genéticos
  - Programação Genética
  - Neuro-Evolução
  - Evolução Diferencial

- Os algoritmos evolucionários funcionam mantendo uma **população de estruturas** que **evoluem** de forma semelhante à evolução das espécies.

Algoritmos Evolucionários

- Nestas estruturas são aplicados **operadores genéticos**, como a **recombinação** e **mutação**.

- Cada indivíduo recebe uma **avaliação (fitness)** que é uma quantificação da sua **qualidade** como solução do problema em questão

- Baseados nesta avaliação são aplicados operadores genéticos de forma a simular a **sobrevivência do mais apto**.

Algoritmos Evolucionários

- Algoritmos evolucionários **buscam** (dentro da atual população) aquelas **soluções** que possuem as **melhores características** e tenta combiná-las de forma a gerar soluções ainda melhores.

- O processo é repetido até que tenha se passado tempo suficiente ou que tenhamos obtido uma solução satisfatória para nosso problema.

Algoritmos Evolucionários

- Algoritmos evolucionários são extremamente dependente de **fatores estocásticos** (probabilísticos), tanto na fase de **inicialização** da população quanto na fase de **evolução**.

- Isto faz com que os seus **resultados raramente sejam perfeitamente reprodutíveis**.

- Além disso, claramente os algoritmos evolucionários são **heurísticas** que não garantem a obtenção do melhor resultado possível em todas as suas execuções.

Algoritmos Evolucionários

- **Conclusão:** se você tem um algoritmo com tempo de execução razoável para solução de um problema, então não há nenhuma necessidade de se usar um algoritmo evolucionário.

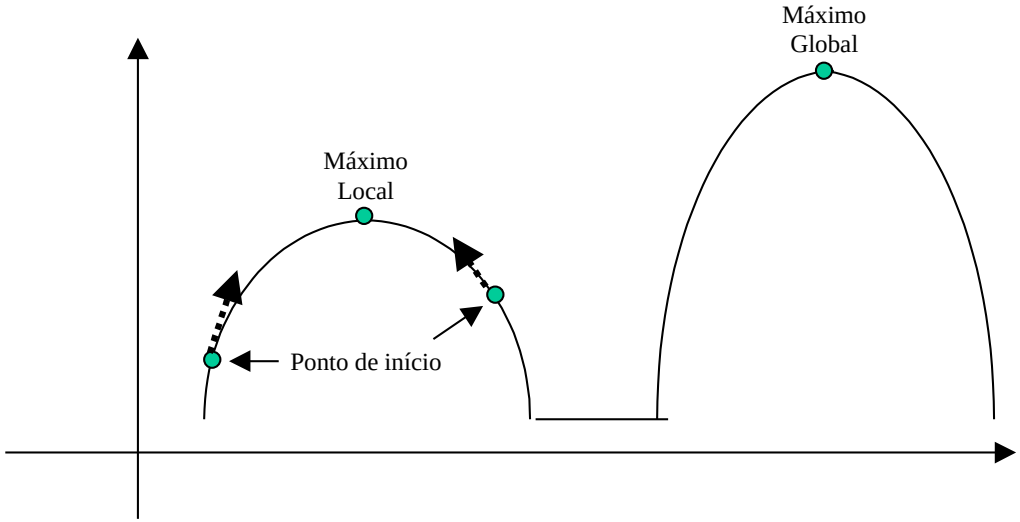
- **Sempre dê prioridade aos algoritmos exatos.**

- Os algoritmos evolucionários entram em cena para resolver aqueles problemas cujos algoritmos exatos são extremamente lentos ou incapazes de obter uma solução.

Algoritmos Genéticos

- **Algoritmos Genéticos** são uma sub-área dos Algoritmos Evolucionários. Logo, são uma metáfora para a evolução natural.

- Os algoritmos genéticos são técnicas heurísticas de otimização global. Com isto, raramente eles ficam presos em **máximos locais**.



Algoritmos Genéticos

- Nos algoritmos genéticos as populações de indivíduos são criadas e submetidas a **operadores genéticos**.
  - Seleção.
  - Recombinação.
  - Mutação.

- Estes operadores utilizam uma caracterização da **qualidade de cada indivíduo** como solução do problema em questão chamada de avaliação do indivíduo (**fitness**).

- É gerado um processo de **evolução natural** destes indivíduos.

Algoritmos Genéticos

- **Definição de um problema** em algoritmos genéticos:
  - É necessário definir uma maneira de codificar os **indivíduos**.
  - Definir os **operadores genéticos** que serão utilizados.
  - Definir uma **função de avaliação** para medir a capacidade de sobrevivência de cada indivíduo.

Algoritmos Genéticos

- **Processo**

1) Inicialize a população de indivíduos;

2) Avalie cada indivíduos na população;

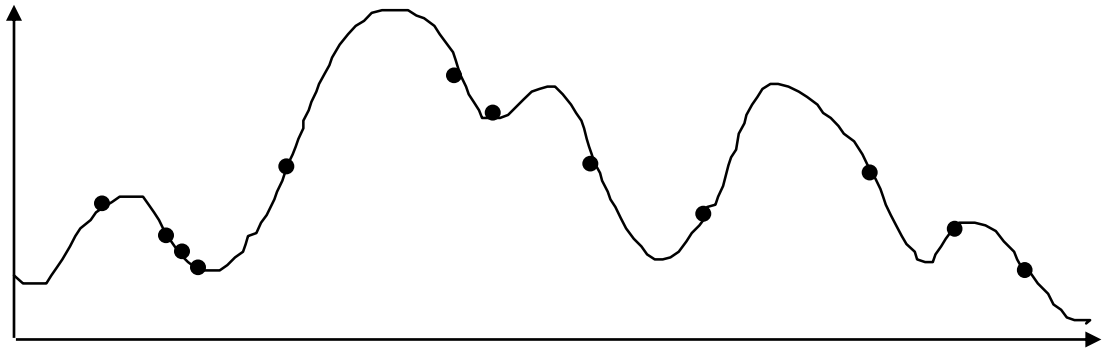
3) Selecione os melhores pais para gerar novos indivíduos. Aplique os operadores de recombinação e mutação a estes pais de forma a gerar os indivíduos da nova geração;

4) Apague os velhos membros da população;

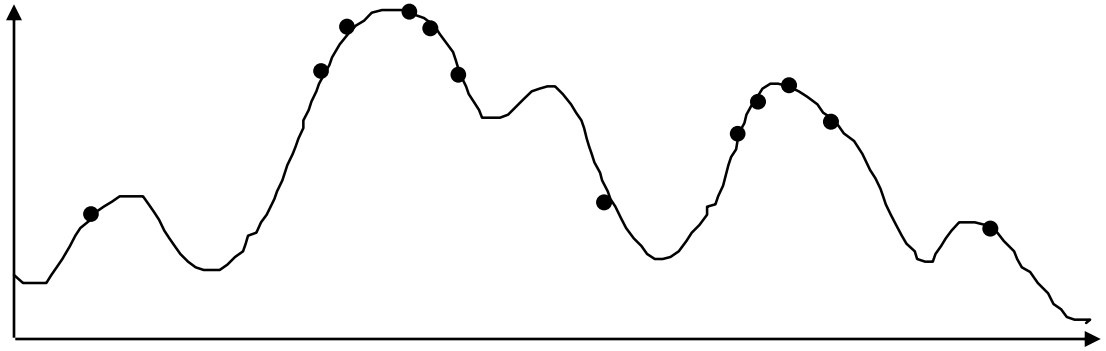
5) Avalie todos os novos indivíduos e insira-os na população;

6) Se o tempo acabou, ou o melhor indivíduos satisfaz os requerimentos da solução do problema, retorne-o, caso contrário volte para o passo 3.

Algoritmos Genéticos



Distribuição dos indivíduos na Geração 0



Distribuição dos indivíduos na Geração N

Algoritmos Genéticos

- Para criar um algoritmo genéticos é necessário:
  - Definir uma maneira de codificar a **população de indivíduos**.
  - Definir uma **função de avaliação**.
  - Definir um método de **seleção dos pais**.
  - Definir os **operadores genéticos**:
    - Recombinação.
    - Mutação.

Codificação da População

- A **representação dos cromossomos** é fundamental para o codificação do algoritmo genético.

- Consiste em uma maneira de traduzir a informação do problema em uma maneira viável de ser tratada pelo computador.

- Cada pedaço indivisível desta representação é chamado de um **gene**, por analogia aos genes que compõem um cromossomo biológico.

Codificação da População

- É importante notar que a representação computacional dos cromossomos é **completamente arbitrária**.

- **Cromossomos podem ser**:
  - Strings de bits (0101 ... 1100)
  - Números reais (43.2 -33.1 ... 0.0 89.2)
  - Listas de regras (R1 R2 R3 ... R22 R23)
  - Qualquer estrutura de dados imaginável!

Exemplo - População

- **Objetivo:** Encontrar o máximo da função $f(x)=x^2$ no intervalo [0,31].

- Os indivíduos da população precisam armazenar o valor de uma **variável inteira**.

- Podemos **codificar cada indivíduo da população** como uma sequência de **5 bits**.

1	0	1	0	0

x=20

0	0	0	1	1

x=3

Função de Avaliação

- A **função de avaliação** é a maneira utilizada pelos algoritmos genéticos para determinar a **qualidade de um indivíduo** como solução do problema em questão.

- **A função de avaliação deve ser escolhida cuidadosamente.** Ela deve embutir todo o conhecimento que se possui sobre o problema a ser resolvido.

Exemplo - Função de Avaliação

- **Objetivo:** Encontrar o máximo da função $f(x)=x^2$ no intervalo [0,31].

- A função de avaliação para este caso consiste simplesmente em converter o número de binário para inteiro e depois elevá-lo ao quadrado.

- Indivíduos que tiverem maiores valores na função de avaliação são os mais aptos.

Seleção dos Pais

- O método de **seleção de pais** deve tentar simular o **mecanismo de seleção natural** que atua sobre as espécies biológicas.
  - Os pais mais capazes geram mais filhos, porém os menos aptos também podem gerar descendentes.
- Temos que privilegiar os indivíduos com função de avaliação alta, sem desprezar completamente aqueles indivíduos com função de avaliação extremamente baixa.
- Isto ocorre pois até indivíduos com péssima avaliação podem ter características genéticas que sejam favoráveis à criação de um "**super indivíduo**".

Seleção dos Pais

- Método mais comum de seleção de pais: **Roleta.**

- Cria-se uma roleta (virtual) na qual cada cromossomo recebe um pedaço proporcional à sua avaliação.

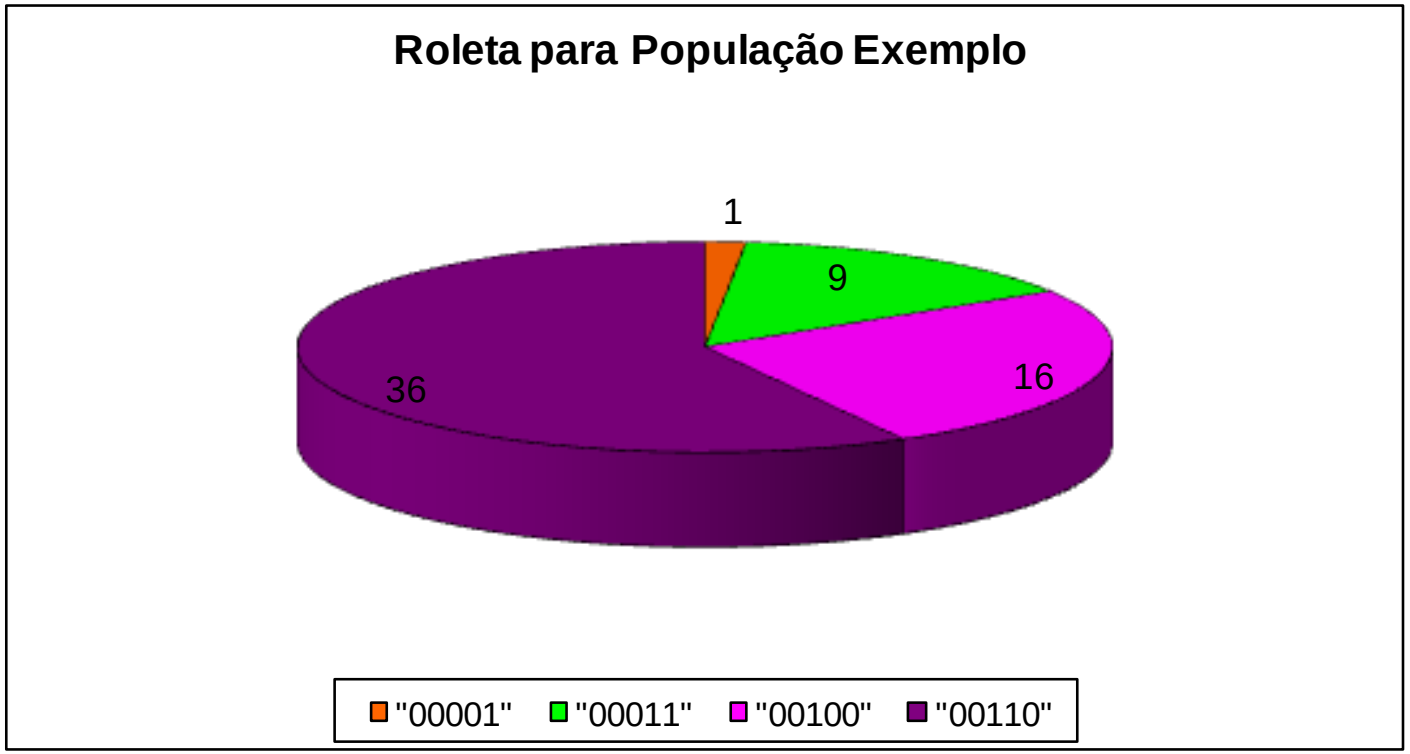
- Roda-se a roleta para sortear os indivíduo que serão pais de um novo indivíduo.

Exemplo - Seleção dos Pais

- Considerando a seguinte população gerada aleatoriamente para o problema de maximização de $f(x)=x^2$ no intervalo [0,31]

Indivíduo	Avaliação	Pedaço da roleta (%)	Pedaço da roleta (º)
00001 (1)	1	1.61	5.8
00011 (3)	9	14.51	52.2
00100 (4)	16	25.81	92.9
00110 (6)	36	58.07	209.1
Total	62	100.00	360.0

Exemplo - Seleção dos Pais



Operadores Genéticos - Recombinação

- **Operador de recombinação de um ponto.**

- **Processo**:

  (1) Seleciona-se **dois pais** através processo de seleção de pais.

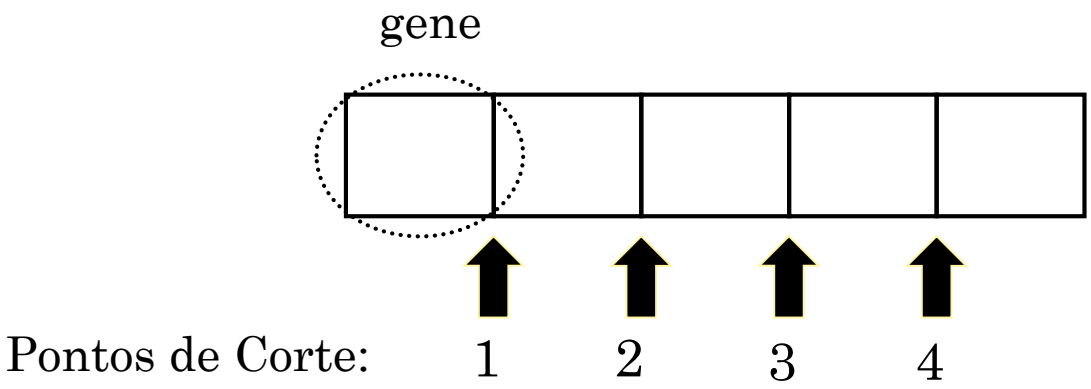
  (2) Seleciona-se **um ponto de corte** (uma posição entre dois genes de um cromossomo). Este ponto de corte é o ponto de separação entre cada um dos genes que compõem o material genético de cada pai.

  (3) A metade à esquerda do ponto de corte vai para um filho e a metade à direita vai para outro.

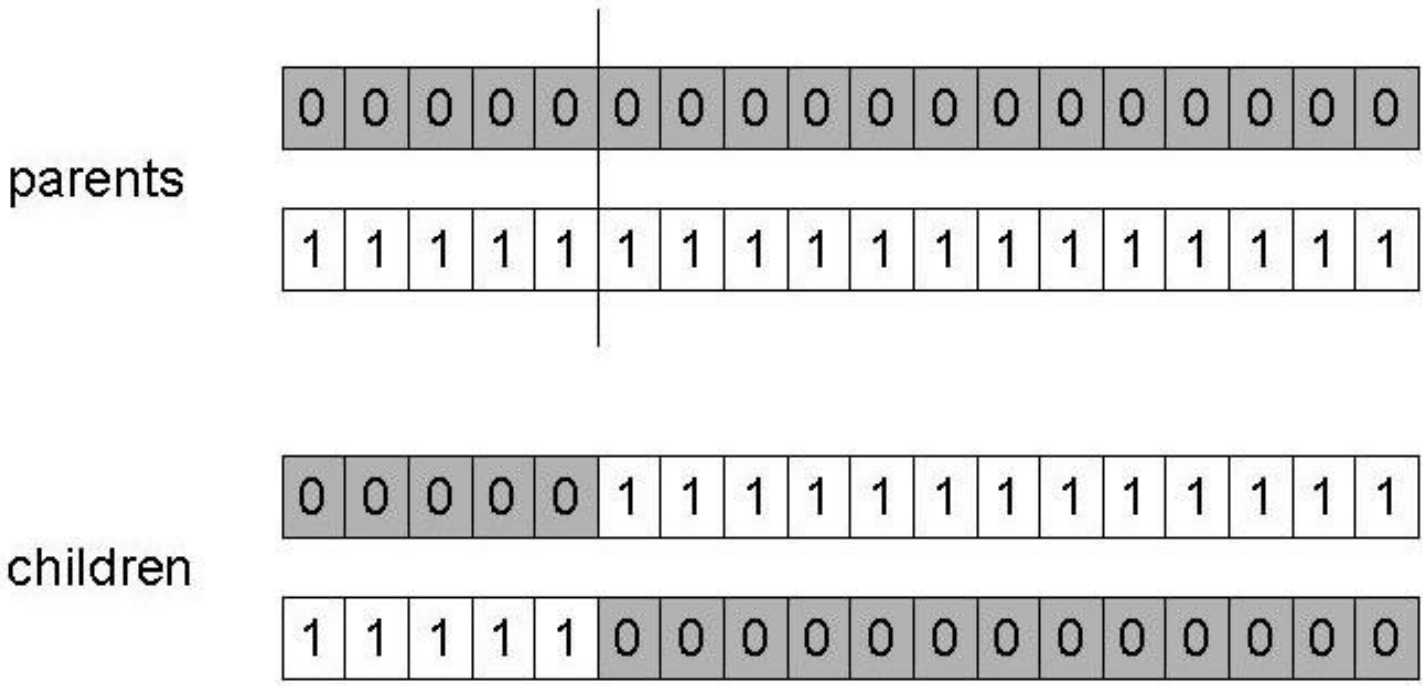
Recombinação - Ponto de Corte

- Cada indivíduo com n genes possui n-1 pontos de corte.

- Em um indivíduo com codificação binária, cada bit é um gene.



Exemplo - Recombinação



Operadores Genéticos - Mutação

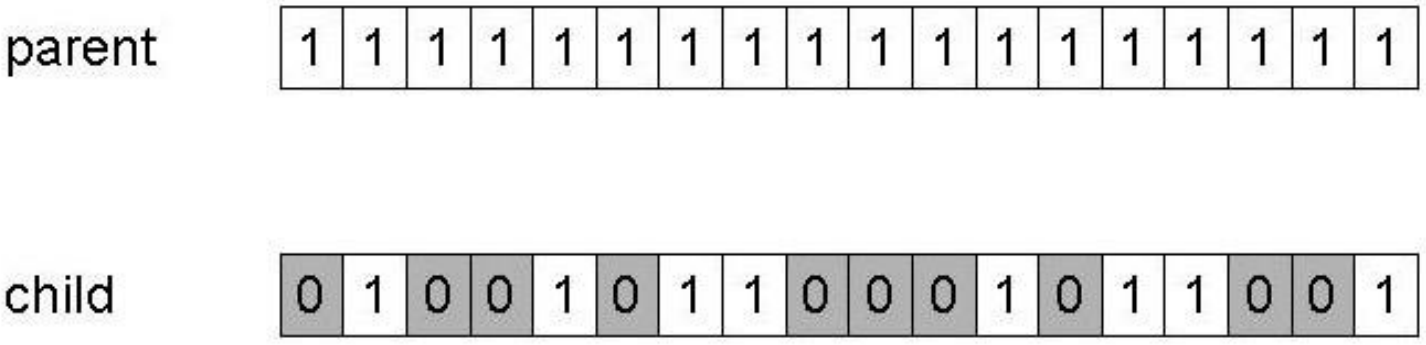
- Depois de compostos os filhos, entra em ação o operador de **mutação**.

- O operador atua com base em uma **probabilidade** extremamente baixa (da ordem de 5%) de **alteração aleatória** do valor de um gene ou mais genes dos filhos.

- O valor da probabilidade que decide se o operador de mutação será ou não aplicado é um dos parâmetros do algoritmo genético que pode alterar o resultado alcançado pelo algoritmo.

Exemplo – Mutação

- Altera-se cada gene de forma independente com base em uma probabilidade p_m
- p_m é denominada taxa de mutação e costuma ser bem baixa.



Outras Técnicas

- Recombinação de mais pontos.
- Recombinação uniforme.
- Elitismo.

Operadores Genéticos

- É possível aumentar ou diminuir a **incidência de cada um dos operadores** sobre a população e assim ter mais controle sobre o desenvolvimento dos cromossomos.

- Cada operador pode receber uma avaliação. Normalmente o operador de recombinação recebe um fitness bem maior que o operador de mutação.

Operadores Genéticos

- **As porcentagem de aplicação de cada operador não precisa ser fixa.**

- No **início** queremos executar **muita reprodução e pouca mutação**, visto que há muita diversidade genética e queremos explorar o máximo possível nosso espaço de soluções.

- Depois de um grande número de gerações, há pouca diversidade genética na população e seria extremamente interessante que o operador de mutação fosse escolhido mais frequentemente.

Recombinação de Dois Pontos

- Existem indivíduos que não podem ser gerados com a recombinação de somente um ponto. Exemplo: 1******1.

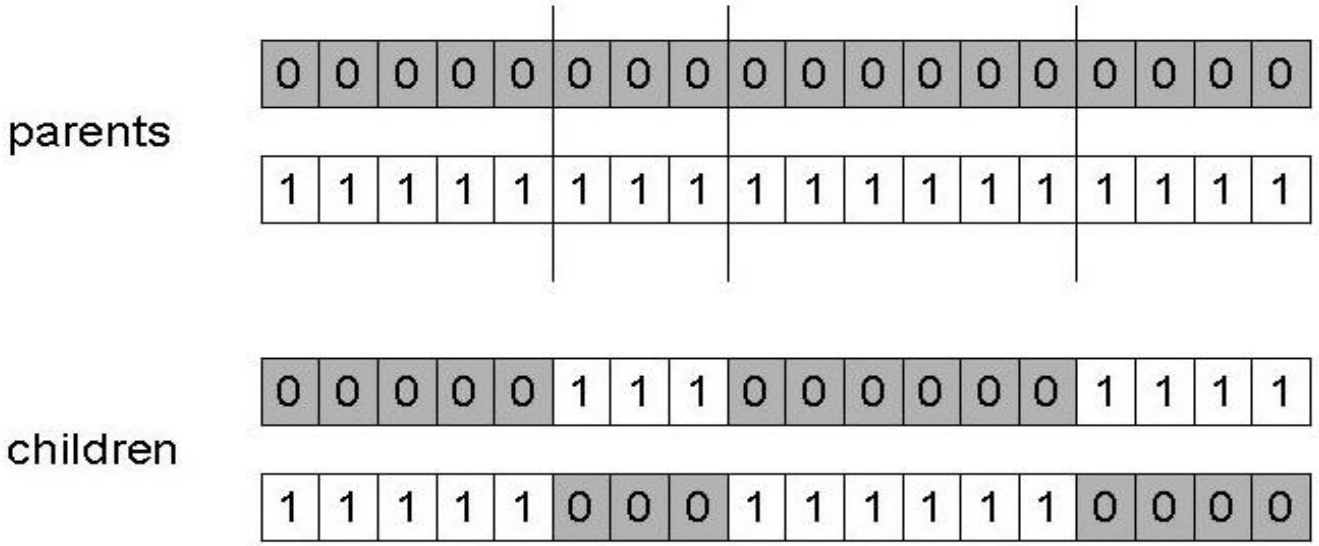
- Consequentemente, se não mudarmos o operador de recombinação, o algoritmo genético fica limitado na sua capacidade de gerar um certo conjunto de cromossomos.

- Para melhorar essa capacidade é possível introduzir a recombinação de 2 pontos.

- Nele, em vez de sortearmos um só ponto de corte, sorteamos dois.

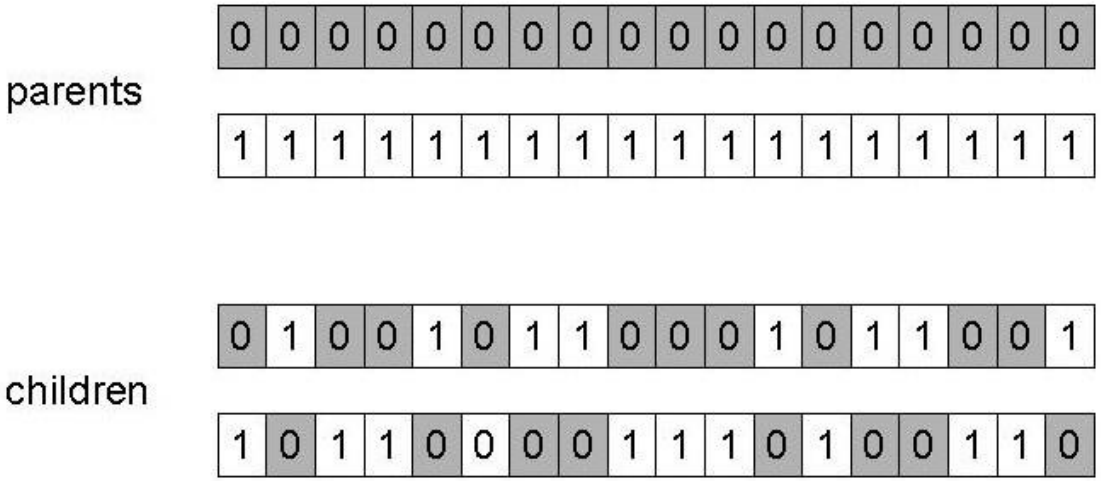
Recombinação de n Pontos

- Evoluindo a ideia da recombinação de dois pontos, é possível tonar o operador uma recombinação de n pontos.



Recombinação Uniforme

- Para cada gene é sorteado um número zero ou um.
  - Se o sorteado for 1, um filho recebe o gene do primeiro pai e o segundo filho o gene do segundo pai.
  - Se o sorteado for 0, o primeiro filho recebe o gene do segundo pai e o segundo filho recebe o gene do primeiro pai.



Elitismo

- A ideia básica por trás do elitismo é a seguinte:
  - Os n melhores indivíduos de cada geração não devem "morrer" junto com a sua geração, mas sim passar para a próxima geração para garantir que seus genomas sejam preservado.
- **É uma forma de garantir que o algoritmo nunca regrida.**