EAD

EDUCAÇÃO A DISTÂNCIA

UNIDADE I

Inteligência Artificial

Profa. Dra. Vanessa Lessa

Perguntas importantes

- **O que é inteligência?**
  - A inteligência é um conceito complexo que abrange vários aspectos do pensamento e da capacidade cognitiva. Em geral, é entendida como a capacidade de aprender, compreender e aplicar conhecimento e habilidades para resolver problemas e adaptar-se a novas situações.

- **O que é Inteligência Artificial (IA)?**
  - A Inteligência Artificial (IA) é o ramo da Ciência da Computação que se dedica ao desenvolvimento de algoritmos e técnicas que permitem que as máquinas simulem a inteligência humana.

Fundamentos da Inteligência Artificial

- Diversas áreas forneceram ideias, percepções e técnicas para a área da Inteligência Artificial.
  - Filosofia.
  - Matemática.
  - Economia.
  - Neurociência.
  - Psicologia.
  - Engenharia de Computadores.
  - Teoria de controle e cibernética.
  - Linguística.

Fundamentos da Inteligência Artificial

- **Filosofia**
  - Conceito de inteligência: A filosofia tem debatido sobre o que é inteligência há séculos. Essas discussões forneceram uma base para a definição de Inteligência Artificial e para a avaliação dos sistemas de IA.
  - Raciocínio lógico: A lógica formal foi desenvolvida pela filosofia para estudar o raciocínio lógico e a dedução.
  - Teoria do conhecimento: A filosofia tem debatido sobre como adquirimos conhecimento e como podemos avaliar a validade desse conhecimento.
  - Ética: A filosofia tem debatido sobre questões éticas relacionadas à tecnologia, incluindo a IA.
    - Filosofia da mente: Concentra-se em entender como a mente funciona e como ela se relaciona com o cérebro.
    - Epistemologia: É a teoria do conhecimento, e ela é importante para a Inteligência Artificial porque ajuda a compreender como o conhecimento é adquirido, organizado e utilizado por sistemas de IA.

Fundamentos da Inteligência Artificial

- **Matemática**
- Álgebra linear: Trata de vetores e matrizes, é usada para resolver problemas de otimização em sistemas de IA, incluindo o treinamento de redes neurais.
- Estatística: Trata de coleta, análise e interpretação de dados, e é usada para avaliar a precisão e a confiabilidade dos sistemas de IA, bem como para selecionar e ajustar modelos.
- Teoria da computação: Estuda as propriedades fundamentais dos algoritmos e a complexidade computacional.
    - Teoria dos grafos: Estuda a estrutura e as propriedades de redes de nós e arestas, e é usada para modelar e analisar sistemas complexos, incluindo sistemas de IA.
    - Cálculo: Estuda as propriedades das funções e suas derivadas, é usado para otimizar os parâmetros dos modelos de IA e para resolver problemas de otimização não lineares.

Fundamentos da Inteligência Artificial

- Teoria da informação: Estuda a quantidade, a natureza e a distribuição da informação, é usada para projetar algoritmos de codificação eficientes para sistemas de IA e para medir a complexidade dos problemas.
- Teoria da decisão: Estuda como as pessoas e as organizações tomam decisões, é usada para desenvolver sistemas de IA que possam tomar decisões racionais e eficientes.

Fundamentos da Inteligência Artificial

- **Economia**
- Teoria dos jogos: Estuda as decisões das pessoas e das organizações em situações de incerteza e com vários jogadores, é usada para desenvolver sistemas de IA que possam tomar decisões racionais e eficientes em situações de incerteza e competição.
- Teoria da escolha racional: Estuda como as pessoas e as organizações tomam decisões, é usada para desenvolver sistemas de IA que possam tomar decisões racionais e eficientes.
- Teoria da eficiência: Estuda como alocar recursos de maneira eficiente, é usada para desenvolver sistemas de IA que possam alocar recursos de maneira eficiente.
    - Teoria da informação econômica: Estuda como a informação afeta as decisões econômicas, é usada para desenvolver sistemas de IA que possam gerenciar e processar grandes volumes de dados.
    - Teoria da organização: Estuda como as organizações funcionam, usado para criar sistemas de IA que possam ser integrados.

Fundamentos da Inteligência Artificial

- Microeconomia: Estuda o comportamento dos indivíduos em relação a preços e mercados, é usada para criar sistemas de IA que possam prever e entender o comportamento de mercado.
- Macroeconomia: Estuda a economia global, é usada para criar sistemas de IA que possam prever e entender tendências econômicas.

Fundamentos da Inteligência Artificial

- **Neurociência**
- Modelos de redes neurais: A base para o desenvolvimento dos modelos de redes neurais, que são amplamente utilizados em sistemas de IA para problemas de classificação e aprendizado profundo.
- Compreensão da neuroplasticidade: Tem estudado como o cérebro se adapta e aprende, é fundamental para desenvolver algoritmos de aprendizado de máquina.
- Estudos da percepção: Estuda como o cérebro processa as informações sensoriais, é fundamental para desenvolver sistemas de IA capazes de processar dados sensoriais.
    - Estudos da linguagem: Tem estudado como o cérebro processa a linguagem, é fundamental para desenvolver sistemas de IA capazes de compreender e produzir linguagem natural.
    - Estudos da memória: Estuda como o cérebro armazena e recupera informações, é fundamental para sistemas de IA capazes de armazenar e recuperar informações.

Fundamentos da Inteligência Artificial

- Estudos do raciocínio: Estuda como o cérebro raciocina, é fundamental para sistemas de IA capazes de raciocinar.
- Estudos da atenção: Estuda como o cérebro seleciona e filtra informações, é fundamental para sistemas de IA capazes de selecionar e filtrar informações.
- Estudos do aprendizado: Estuda como o cérebro aprende, é fundamental para sistemas de IA capazes de aprender.

Fundamentos da Inteligência Artificial

- **Psicologia**
  - Compreensão da percepção: Estuda como as pessoas processam informações sensoriais, o que é fundamental para desenvolver sistemas de IA capazes de processar dados sensoriais.
  - Compreensão da linguagem: Estuda como as pessoas processam a linguagem, o que é fundamental para desenvolver sistemas de IA capazes de compreender e produzir linguagem natural.
  - Compreensão da memória: Estuda como as pessoas armazenam e recuperam informações, o que é fundamental para desenvolver sistemas de IA capazes de armazenar e recuperar informações.
    - Compreensão do raciocínio: Estuda como as pessoas raciocinam, o que é fundamental para desenvolver sistemas de IA capazes de raciocinar.

Fundamentos da Inteligência Artificial

- Compreensão da atenção: Estuda como as pessoas selecionam e filtram informações, o que é fundamental para desenvolver sistemas de IA capazes de selecionar e filtrar informações.
- Compreensão do aprendizado: Estuda como as pessoas aprendem, o que é fundamental para desenvolver sistemas de IA capazes de aprender.
- Compreensão da inteligência: Estuda a inteligência, é fundamental para desenvolver sistemas de IA que possam simular a inteligência humana.
- Compreensão da psicologia social: Estuda como as pessoas se relacionam umas com as outras, é fundamental para desenvolver sistemas de IA que possam se relacionar de maneira natural com os humanos.

Fundamentos da Inteligência Artificial

- **Engenharia de Computadores**
  - Desenvolvimento de algoritmos: Desenvolve algoritmos para o aprendizado de máquina, o reconhecimento de padrões, a otimização e outras tarefas fundamentais para a IA.
  - Desenvolvimento de arquiteturas: Desenvolve arquiteturas para sistemas de IA, incluindo arquiteturas de redes neurais, arquiteturas baseadas em regras e arquiteturas híbridas.
  - Desenvolvimento de ferramentas: Para ajudar a desenvolver e implementar sistemas de IA, incluindo ferramentas para o treinamento e validação de modelos, ferramentas para a gestão de dados e ferramentas para a otimização de recursos computacionais.
    - Desenvolvimento de sistemas: Desenvolve sistemas de aprendizado de máquina, sistemas baseados em regras, sistemas de processamento de linguagem natural e sistemas de visão computacional.

Fundamentos da Inteligência Artificial

- Desenvolvimento de infraestrutura: Desenvolve infraestrutura para suportar sistemas de IA, incluindo infraestrutura de nuvem, infraestrutura de *hardware* especializado, infraestrutura de segurança e infraestrutura de gerenciamento de dados.
- Otimização de recursos: Otimizar os recursos para o desenvolvimento de sistemas, como otimização de tempo de processamento, otimização de uso de memória e otimização de uso de banda de dados.
- Integração com outras tecnologias: Estuda meios para integrar sistemas de IA com outras tecnologias, como robótica, Internet das Coisas (IoT), automação e Inteligência Artificial distribuída.

Fundamentos da Inteligência Artificial

- **Teoria de controle e cibernética**
  - Conceitos de *feedback* e controle: Fornece conceitos fundamentais de *feedback* e controle, que são utilizados em sistemas de IA para ajustar o comportamento dos sistemas de acordo com os resultados desejados.
  - Modelos de sistemas dinâmicos: Desenvolve modelos matemáticos de sistemas dinâmicos, que são utilizados em sistemas de IA para prever e controlar o comportamento de sistemas complexos.
  - Controladores baseados em modelos: Desenvolve controladores baseados em modelos, que são utilizados em sistemas de IA para controlar o comportamento de sistemas automatizados.
    - Teoria de sistemas inteligentes: Estuda como criar sistemas que possam aprender, se adaptar e se auto-organizar, é fundamental para desenvolver sistemas de IA que possam aprender e se adaptar.

Fundamentos da Inteligência Artificial

- Teoria de sistemas autônomos: Estuda como criar sistemas que possam tomar decisões e agir de forma independente, é fundamental para criar sistemas de IA que possam tomar decisões e agir de forma autônoma.
- Teoria de sistemas adaptativos: Estuda como criar sistemas que possam se adaptar a mudanças no ambiente, isso é fundamental para desenvolver sistemas de IA que possam se adaptar a mudanças no ambiente.
- Teoria de sistemas inteligentes distribuídos: Estuda como criar sistemas distribuídos que possam trabalhar juntos de forma inteligente. Isso é fundamental para desenvolver sistemas de IA distribuídos que possam trabalhar juntos de forma inteligente.

Fundamentos da Inteligência Artificial

- **Linguística**
- Compreensão da linguagem natural: Estuda como as línguas funcionam, incluindo sua estrutura, sintaxe e semântica, é fundamental para criar sistemas de IA capazes de compreender e produzir linguagem natural.
- Análise de sentimentos: Estuda como os sentimentos são expressos na linguagem, o que é fundamental para desenvolver sistemas de IA capazes de identificar e analisar sentimentos em textos.
- Técnicas de processamento de linguagem natural: Desenvolve técnicas de análise sintática, análise semântica e geração de texto, utilizadas em sistemas de IA para problemas de processamento de linguagem natural.
    - Análise de discurso: Estuda como as pessoas se comunicam, como as estruturas de discurso e os contextos afetam a interpretação, usado para criar sistemas capazes de compreender e produzir textos de forma natural.

Fundamentos da Inteligência Artificial

- Estudos de pragmática: Estuda como as pessoas usam a linguagem para comunicar intenções, inferir significado implícito e lidar com ambiguidade, é fundamental para criar sistemas de IA capazes de compreender e produzir linguagem natural.
- Estudos de semântica e estudos de sintaxe: Estuda como as pessoas usam a linguagem para comunicar significado, incluindo como as palavras e as frases são relacionadas e como as palavras são relacionadas com o mundo, o que é fundamental para desenvolver sistemas de IA capazes de compreender e produzir linguagem natural.

A evolução da Inteligência Artificial

- A gestação da Inteligência Artificial (1943-1955).
- O nascimento da Inteligência Artificial (1956).
- Entusiasmo inicial, grandes expectativas (1952-1969).
- Uma dose de realidade (1966-1973).
- Sistemas baseados em conhecimento: a chave para o poder? (1969-1979).
- A IA se torna uma indústria (de 1980 até a atualidade).
- O retorno das redes neurais (de 1986 até a atualidade).
- A IA se torna uma ciência (de 1987 até a atualidade).
    - O surgimento de agentes inteligentes (de 1995 até a atualidade).
    - Disponibilidade de conjuntos de dados muito grandes (2001 até a atualidade).

Pesquisas realizadas

- Veículos robóticos.
- Reconhecimento de voz.
- Planejamento autônomo e escalonamento.
- Jogos.
- Combate a *spam*.
- Planejamento logístico.
- Robótica.
- Tradução automática.

Softwares inteligentes

- Os *softwares* inteligentes que usam Inteligência Artificial são programas que foram projetados para realizar tarefas que normalmente requerem inteligência humana.
- Um exemplo de *software* inteligente é o **assistente virtual**, como o Amazon Alexa ou o Google Assistant. Esses programas usam a IA para entender e responder às perguntas dos usuários em linguagem natural, além de realizar tarefas como reproduzir música, controlar dispositivos domésticos inteligentes e fornecer informações sobre o tempo.
- O *software* de **reconhecimento de imagens**, como o usado em sistemas de segurança de câmeras ou em aplicativos de reconhecimento facial. Esses programas usam técnicas de aprendizado profundo para reconhecer rostos e outros objetos em imagens, permitindo que as máquinas tomem decisões baseadas nessas informações.

Softwares inteligentes

- O **chatbot** é outro exemplo de *software* inteligente que usa IA. São projetados para conversar com os usuários em linguagem natural e podem ser usados para atender a perguntas de clientes, ajudar a navegar em um *site* ou fornecer suporte técnico.
- A Inteligência Artificial está sendo usada em *softwares* de análise de dados, como os sistemas de **Business Intelligence (BI)**, que ajudam as empresas a entender e tomar decisões com base em grandes conjuntos de dados.

Interatividade

Analise as afirmações:

I. A inteligência é um conceito simples que abrange apenas aspectos do pensamento.
II. A Inteligência Artificial (IA) é o ramo da Ciência da Computação que se dedica ao desenvolvimento de algoritmos e técnicas que permitem que as máquinas simulem a inteligência humana.
III. Apenas a área da Ciência da Computação fornece ideias, percepções e técnicas para a área da Inteligência Artificial.
IV. Os *softwares* inteligentes que usam Inteligência Artificial são programas que foram projetados para realizar tarefas que normalmente requerem inteligência humana.

Está(ão) correta(s):

a) I e II .
b) I e III.
c) I e IV.
d) II e III.
e) II e IV.

Resposta

Analise as afirmações:

I. A inteligência é um conceito simples que abrange apenas aspectos do pensamento.
II. A Inteligência Artificial (IA) é o ramo da Ciência da Computação que se dedica ao desenvolvimento de algoritmos e técnicas que permitem que as máquinas simulem a inteligência humana.
III. Apenas a área da Ciência da Computação fornece ideias, percepções e técnicas para a área da Inteligência Artificial.
IV. Os *softwares* inteligentes que usam Inteligência Artificial são programas que foram projetados para realizar tarefas que normalmente requerem inteligência humana.

Está(ão) correta(s):

a) I e II .
b) I e III.
c) I e IV.
d) II e III.
e) II e IV.

Agentes inteligentes

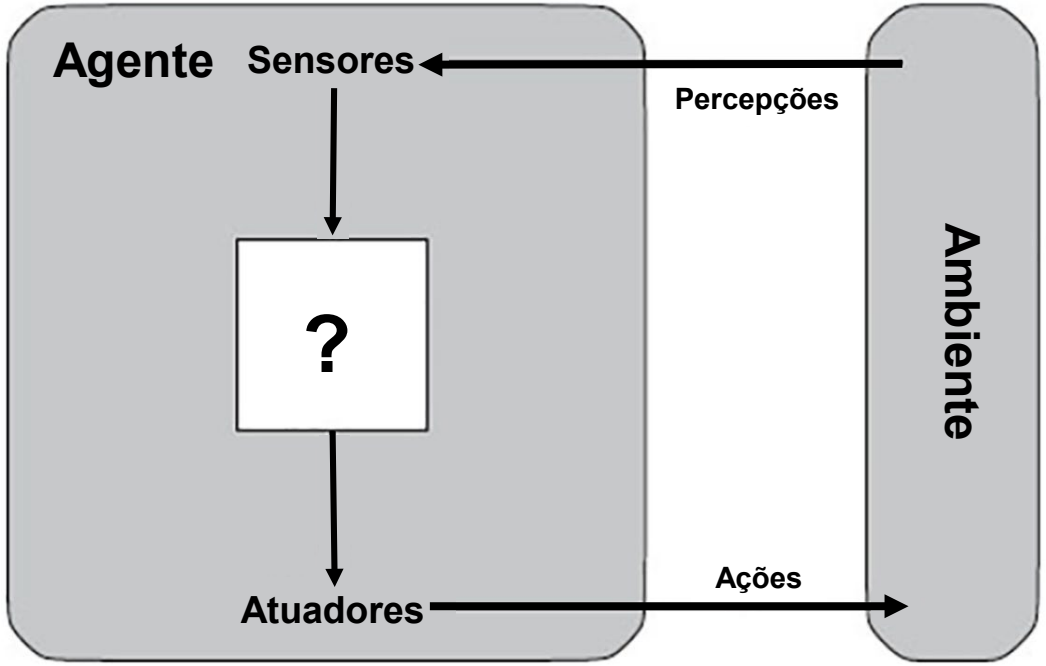
- Um agente inteligente é uma entidade capaz de perceber o seu ambiente, tomar decisões e realizar ações com o objetivo de atingir metas específicas. Esses agentes são projetados para funcionar autonomamente e aprender a partir de sua interação com o ambiente, a fim de melhorar sua *performance* ao longo do tempo.

- Um agente percebe o ambiente por meio de sensores e outras formas de entrada de dados. Esses dados são processados e interpretados pelo agente para produzir uma representação interna do estado atual do ambiente. O agente pode usar essa representação para tomar decisões sobre as ações a serem realizadas a fim de atingir seus objetivos.

Estrutura dos agentes

Os agentes são entidades que tomam ações no ambiente para alcançar objetivos específicos. A estrutura de um agente geralmente inclui:

- Sensores: que permitem ao agente perceber o ambiente ao seu redor.
- Ações: que permitem ao agente interagir com o ambiente.
- Uma lógica de tomada de decisão: que permite ao agente escolher a ação mais adequada para alcançar seu objetivo.
- Um objetivo ou metas: que definem o que o agente está tentando alcançar.



Fonte: Adaptado de: NORVIG (2013, p. 31).

Especificar as PEAS

PEAS é um acrônimo que representa um modelo de quatro componentes que descreve a especificação de um agente inteligente. Os quatro componentes são:

- P (*Performance*): a medida de desempenho usada para avaliar a eficácia do agente em realizar uma tarefa. A medida de desempenho deve estar relacionada aos objetivos da tarefa e pode incluir métricas como tempo de resposta, precisão, economia, entre outros.
- E (*Environment*): o ambiente em que o agente opera e interage. O ambiente pode ser físico, virtual ou ambos, e pode incluir vários elementos como objetos, outros agentes, obstáculos, informações, entre outros.
    - A (*Actuators*): os atuadores são os mecanismos que permitem que o agente realize ações no ambiente. Os atuadores podem incluir dispositivos físicos, como braços robóticos, bem como *softwares* que controlam a interação com o ambiente.
    - S (*Sensors*): os sensores são os mecanismos que permitem que o agente perceba o ambiente e receba informações relevantes para realizar ações. Os sensores podem incluir câmeras, sensores de temperatura, entre outros.

Tipos de agentes

- O trabalho da IA é desenvolver o programa do agente implementando funções que mapeiam percepções em ação. O programa que desenvolvemos deve estar de acordo com a arquitetura do agente (considerando um dispositivo de computador com sensores e atuadores físicos), dessa forma, se o programa sugerir a ação de CAMINHAR, a arquitetura deve estar preparada para essa ação.

Agente = Arquitetura + Programa

Descreveremos quatro tipos de programas de agentes:

- Agentes reativos simples.
- Agentes reativos baseados em modelo.
- Agentes baseados em objetivos.
- Agentes baseados na utilidade.

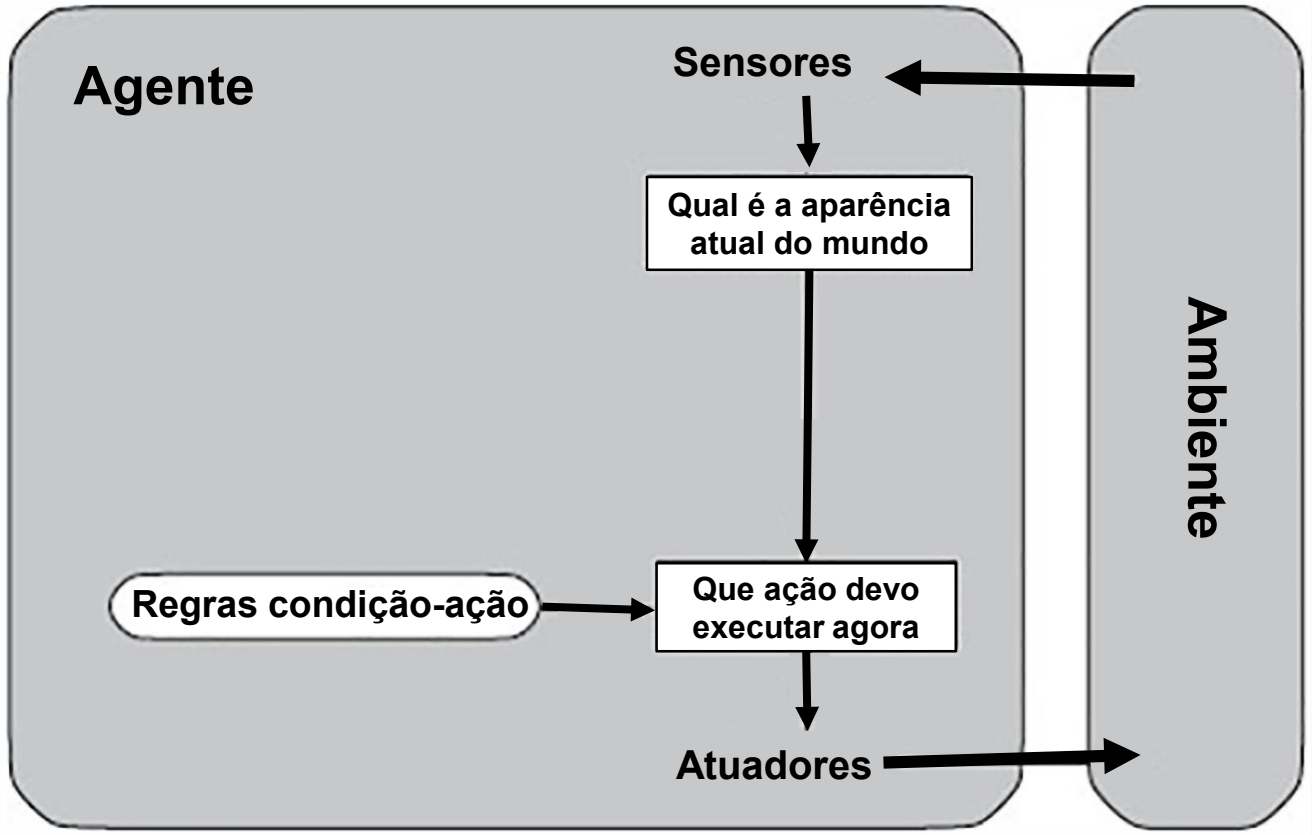
Agentes reativos simples

- Agentes reativos simples são uma subclasse de agentes que tomam decisões baseadas apenas na percepção atual, sem se preocupar com o estado anterior do ambiente. Esses agentes geralmente possuem uma estrutura simples, que inclui sensores, ações e uma lógica de tomada de decisão simples. Eles não possuem um estado interno, ou seja, não armazenam informações sobre o ambiente anterior e nem tem a capacidade de planejar ações futuras baseadas em objetivos.
- Agentes reativos simples são usados em aplicações em que a complexidade do ambiente é baixa e as ações do agente são baseadas principalmente na percepção atual. Eles são comumente utilizados em sistemas de robótica simples, como robôs de limpeza.
    - Por serem simples, esses agentes possuem uma baixa complexidade de tempo e espaço, ou seja, são rápidos e não precisam de muita memória para funcionar.

Agentes reativos simples

função AGENTE-REATIVO-SIMPLES (percepção) **retorna** uma ação

variáveis estáticas: regras, um conjunto de regras condição-ação
estado ← INTERPRETAR-ENTRADA (percepção)
regra ← REGRA-CORRESPONDENTE (estado, regras)
ação ← AÇÃO-DA-REGRA [regra]
retornar ação



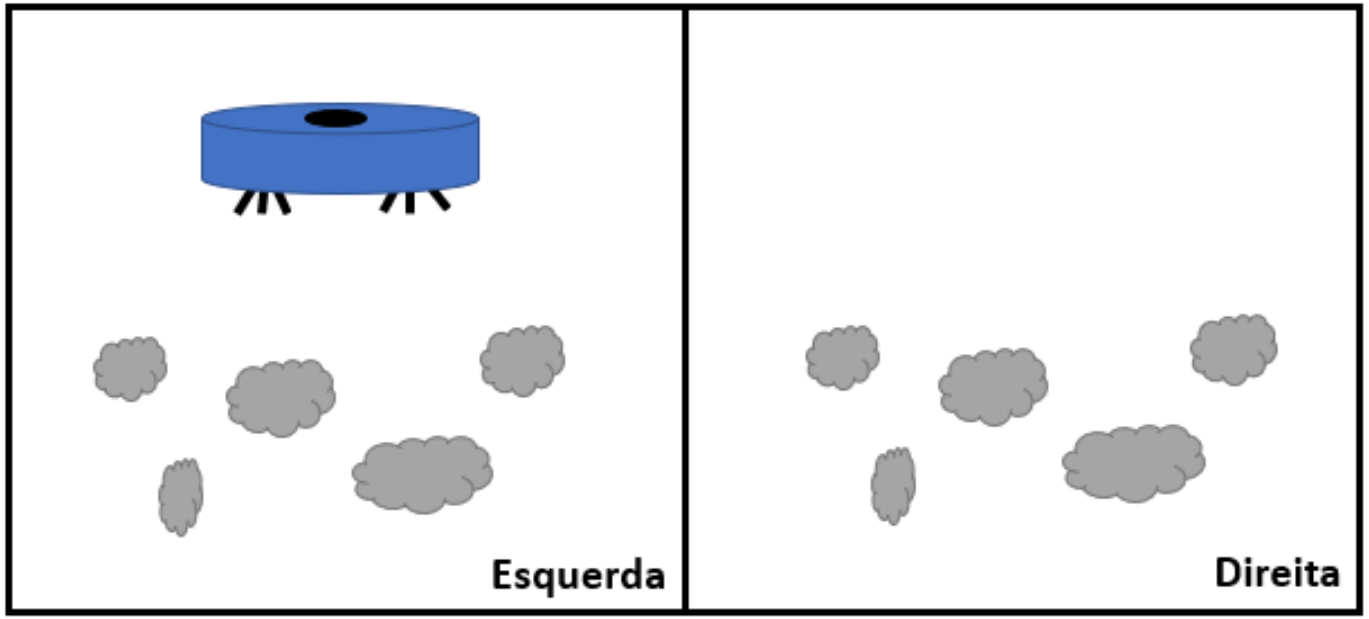
Fonte: Adaptado de: NORVIG (2013, p. 43).

Agentes reativos simples

função AGENTE-ASPIRADOR-DE-PÓ-REATIVO ([sala, situação]) **retorna** uma ação*

se situação = Sujo **então** retorna Aspirar
senão se sala = Esquerda **então retorna** Direita
senão se sala = Direita **então retorna** Esquerda

Fonte: NORVIG (2013, p. 43).



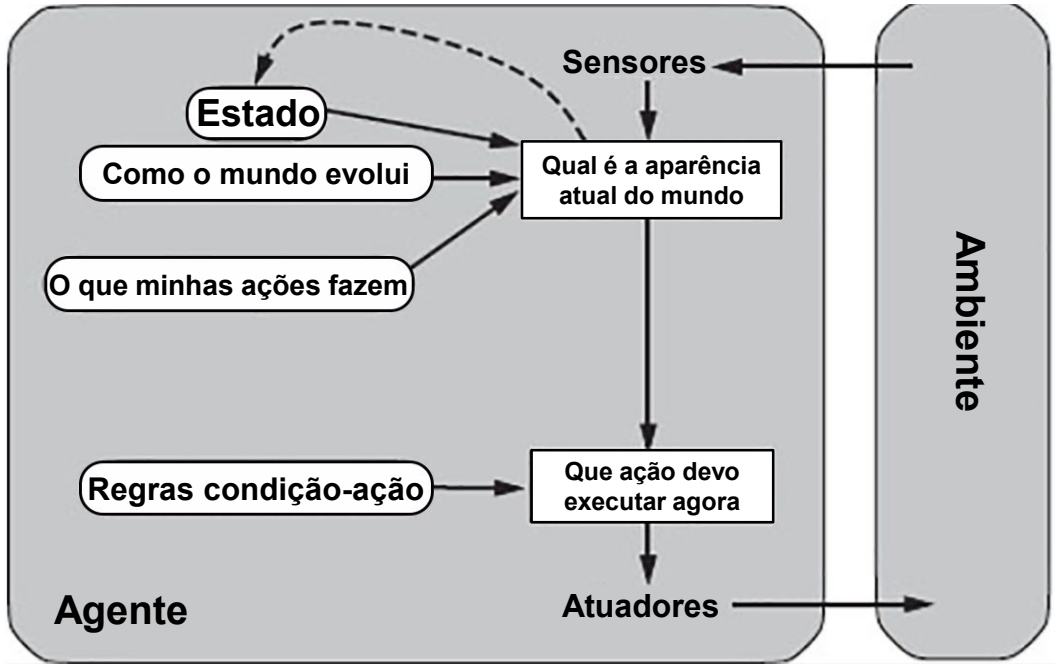
Fonte: Autoria própria.

Agentes reativos baseados em modelo

- Agentes reativos baseados em modelo são uma subclasse de agentes que, além de tomar decisões baseadas apenas na percepção atual, também mantêm um modelo interno do ambiente. Eles armazenam informações sobre o estado anterior do ambiente e utilizam essas informações para tomar decisões futuras. Dessa forma, eles são capazes de lidar com situações incertas e imprevisíveis, já que possuem um modelo interno do ambiente.
- Os agentes reativos baseados em modelo são usados em aplicações em que a complexidade do ambiente é moderada e as ações do agente são baseadas tanto na percepção atual quanto no estado anterior do ambiente. Eles são comumente utilizados em sistemas de robótica avançados, como robôs autônomos, e em jogos mais complexos.
    - A complexidade computacional dos agentes reativos baseados em modelo é geralmente maior do que a dos agentes reativos simples, pois eles precisam manter e atualizar o modelo interno do ambiente.

Agentes reativos baseados em modelo

função AGENTE-REATIVO-BASEADO-EM-MODELOS (percepção) retorna uma ação
persistente: estado, a concepção do agente do estado atual do mundo
modelo, uma descrição de como o próximo estado depende do estado atual e da ação
regras, um conjunto de regras condição-ação
ação, a ação mais recente, inicialmente nenhuma
estado ← ATUALIZAR-ESTADO (estado, ação, percepção, modelo)
regra ← REGRA-CORRESPONDENTE (estado, regras)
ação ← regra, AÇÃO
retornar ação

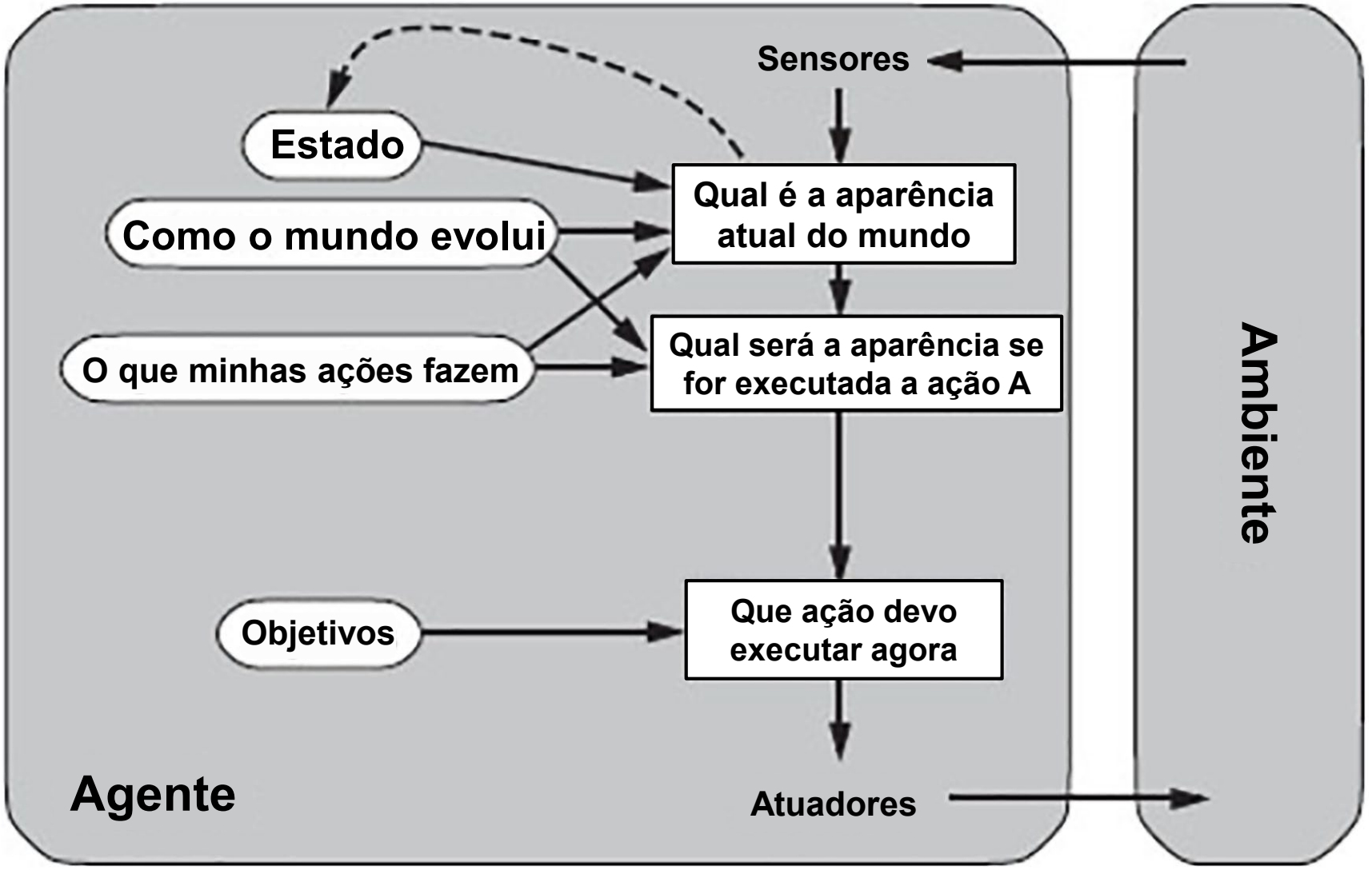


Fonte: Adaptado de: NORVIG (2013, p. 44).

Agentes baseados em objetivos

- Agentes baseados em objetivos são uma subclasse de agentes que possuem metas ou objetivos específicos a serem alcançados. Esses agentes possuem uma estrutura mais complexa do que os agentes reativos simples ou baseados em modelo, incluindo sensores, ações, uma lógica de tomada de decisão avançada e uma capacidade de planejar ações futuras baseadas em objetivos.
- Os agentes baseados em objetivos são usados em aplicações em que a complexidade do ambiente é alta e as ações do agente são baseadas tanto na percepção atual quanto no estado anterior do ambiente, além de objetivos futuros a serem alcançados. Eles são comumente utilizados em aplicações avançadas como Inteligência Artificial para jogos, robótica autônoma, assistentes virtuais, entre outros.
    - A complexidade computacional dos agentes baseados em objetivos é geralmente maior do que a dos agentes reativos simples ou baseados em modelo, pois eles precisam planejar ações futuras baseadas em objetivos e lidar com situações incertas e imprevisíveis.

Agentes baseados em objetivos

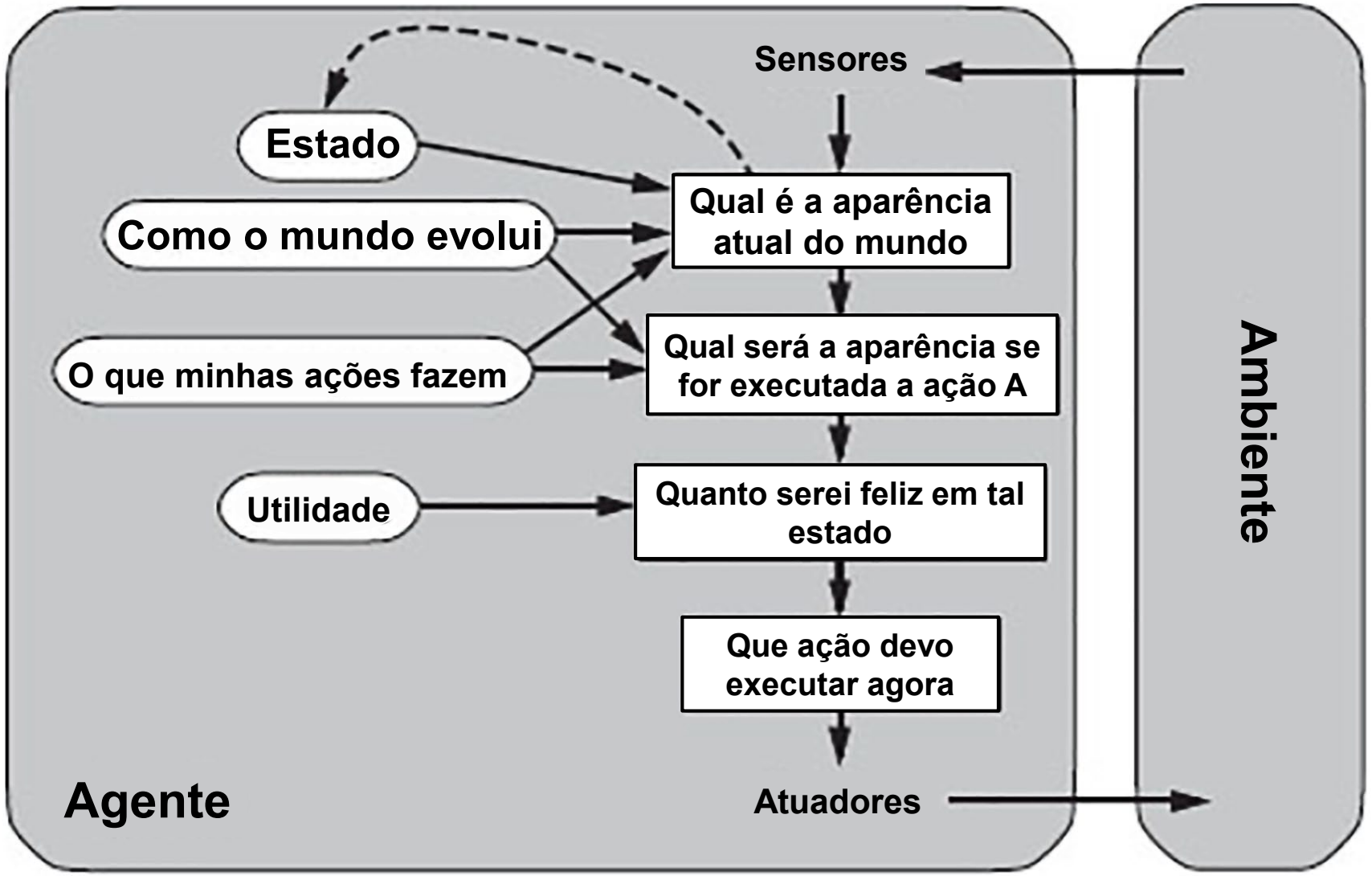


Fonte: Adaptado de: NORVIG (2013, p. 45).

Agentes baseados em utilidade

- Agentes baseados em utilidade são uma subclasse de agentes baseados em objetivos que utilizam a teoria da utilidade para tomar decisões. Essa teoria se baseia na ideia de que a escolha de uma ação deve ser baseada na utilidade esperada dessa ação, ou seja, no valor esperado de alcançar o objetivo.
- Para calcular a utilidade de uma ação, os agentes baseados em utilidade usam uma função de utilidade, que atribui um valor numérico a cada estado possível do ambiente. Essa função é geralmente definida pelo *designer* do sistema ou aprendida a partir de dados de treinamento.
    - Os agentes baseados em utilidade são usados em aplicações em que a complexidade do ambiente é alta e as ações do agente são baseadas tanto na percepção atual quanto no estado anterior do ambiente, além de objetivos futuros a serem alcançados e a utilidade dessas ações.
    - A complexidade computacional dos agentes baseados em utilidade é geralmente maior do que a dos agentes baseados em objetivos, pois eles precisam calcular a utilidade esperada das ações e lidar com as situações.

Agentes baseados em utilidade



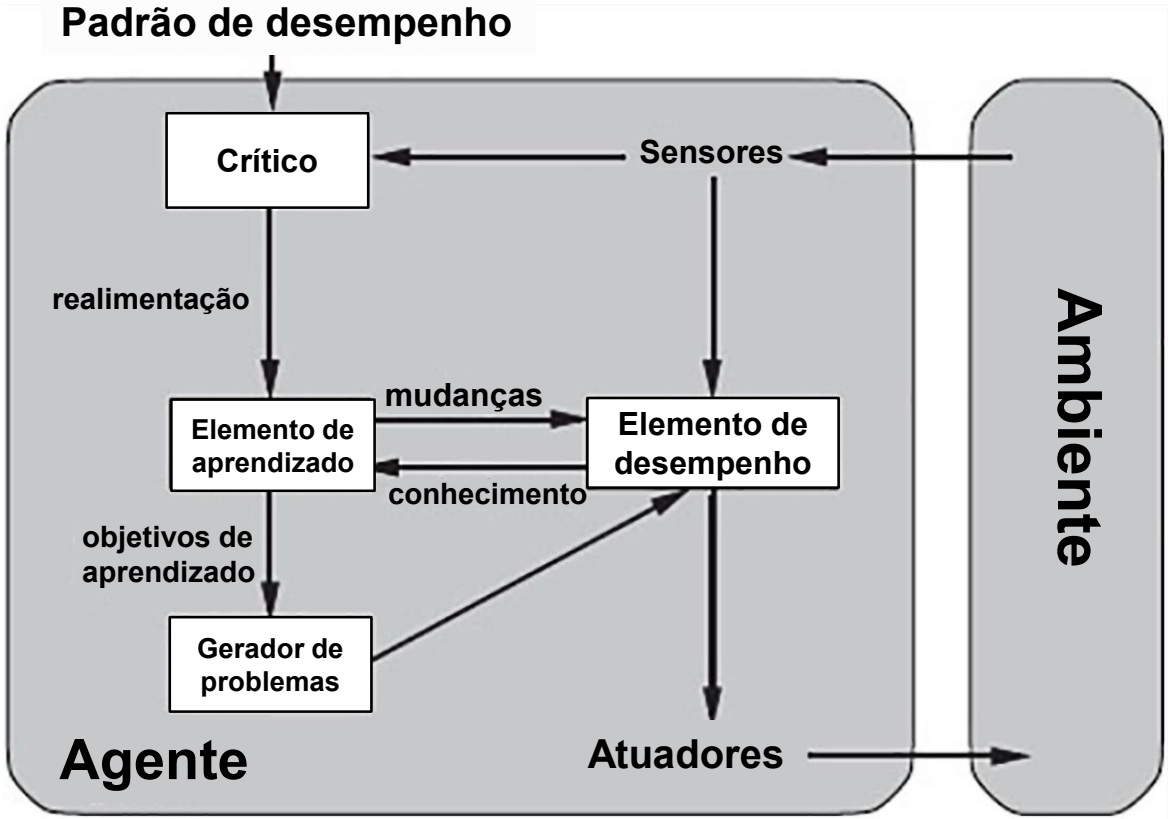
Fonte: Adaptado de: NORVIG (2013, p. 46).

Agentes com aprendizagem

- Agentes com aprendizagem são uma subclasse de agentes que são capazes de aprender e melhorar suas ações ao longo do tempo. Eles usam algoritmos de aprendizado de máquina para ajustar suas ações baseadas em *feedback* de seu desempenho. Dessa forma, eles podem adaptar-se a mudanças no ambiente e melhorar sua capacidade de alcançar seus objetivos.
- Os agentes com aprendizagem podem ser classificados como supervisionados, não supervisionados ou reforço.
    - **Agentes supervisionados:** são aqueles que aprendem a partir de dados rotulados, em que o objetivo é aprender uma função que mapeia entradas para saídas.
    - **Agentes não supervisionados:** são aqueles que aprendem a partir de dados não rotulados, em que o objetivo é descobrir padrões ocultos nos dados.
    - **Agentes de reforço:** são aqueles que aprendem por meio da tentativa e erro, recebendo recompensas ou punições para cada ação tomada.

Agentes com aprendizagem

- Elemento de aprendizado: responsável pela execução de aperfeiçoamentos.
- Elemento de desempenho: responsável pela seleção de ações externas.
- Crítico: informa ao elemento de aprendizado como o agente está se comportando em relação a um padrão fixo de desempenho.
- Gerador de problemas: responsável por sugerir ações que levarão a experiências novas e informativas.



Fonte: Adaptado de: NORVIG (2013, p. 47).

Sistemas multiagentes

- Sistemas multiagentes são sistemas compostos por múltiplos agentes que interagem entre si e com o ambiente. Esses agentes podem ser individuais ou coletivos, e cada um pode ter sua própria estrutura, lógica de tomada de decisão e objetivos.
- Os sistemas multiagentes são usados em aplicações em que a complexidade do ambiente é muito alta e as ações de um único agente são insuficientes para alcançar os objetivos desejados. Ao dividir a tarefa entre vários agentes, é possível resolver problemas mais complexos e lidar com incertezas e imprevisibilidade.
    - Os sistemas multiagentes são utilizados em várias áreas, como Inteligência Artificial para jogos, robótica autônoma, Inteligência Artificial para transporte, Inteligência Artificial para finanças, Inteligência Artificial para saúde, em simulações de tráfego, simulações de sistemas econômicos, simulações de ecossistemas, entre outros.
    - A complexidade computacional dos sistemas multiagentes é geralmente maior do que a dos agentes individuais, pois eles precisam lidar com a interação entre os agentes e a coordenação de suas ações.

Inteligência Artificial Distribuída (IAD)

- É uma abordagem para a Inteligência Artificial que consiste em dividir tarefas complexas entre múltiplos computadores ou dispositivos, que trabalham juntos de forma coordenada para alcançar um objetivo comum. Essa abordagem permite que sistemas de inteligência artificial possam lidar com grandes volumes de dados e processamento intensivo, além de permitir escalabilidade e tolerância a falhas.
- Os sistemas de inteligência artificial distribuídos podem ser classificados em dois tipos: sistemas de inteligência artificial distribuídos centralizados e sistemas de inteligência artificial distribuídos descentralizados.
- Sistemas de inteligência artificial distribuídos centralizados possuem um único ponto central de controle que gerencia a distribuição de tarefas e a comunicação entre os nós.
    - Sistemas de inteligência artificial distribuídos descentralizados não possuem um único ponto central de controle e cada nó age de forma autônoma.

Interatividade

Analise as afirmações:

I. Agentes reativos simples tomam decisões baseadas apenas na percepção atual, sem se preocupar com o estado anterior do ambiente.
II. Agentes reativos baseados em modelo tomam decisões baseadas apenas na percepção atual e também mantêm um modelo interno do ambiente.
III. Agentes baseados em objetivos possuem objetivos específicos a serem alcançados.
IV. Agentes baseados em utilidade utilizam a teoria da utilidade para tomar decisões.
V. Agentes com aprendizagem são capazes de aprender e melhorar suas ações ao longo do tempo.

Estão corretas:

a) I, II e III.
b) I, II e IV.
c) I, II e V.
d) I, III, IV e V.
e) Todas as afirmações.

Resposta

Analise as afirmações:

I. Agentes reativos simples tomam decisões baseadas apenas na percepção atual, sem se preocupar com o estado anterior do ambiente.
II. Agentes reativos baseados em modelo tomam decisões baseadas apenas na percepção atual e também mantêm um modelo interno do ambiente.
III. Agentes baseados em objetivos possuem objetivos específicos a serem alcançados.
IV. Agentes baseados em utilidade utilizam a teoria da utilidade para tomar decisões.
V. Agentes com aprendizagem são capazes de aprender e melhorar suas ações ao longo do tempo.

Estão corretas:

a) I, II e III.
b) I, II e IV.
c) I, II e V.
d) I, III, IV e V.
e) Todas as afirmações.

Resolução de problemas utilizando algoritmos de busca

- A resolução de problemas é o processo de encontrar uma solução para um problema usando técnicas de análise e raciocínio.
- Os algoritmos de busca são um conjunto de técnicas usadas para encontrar soluções para problemas em que é necessário procurar entre muitas opções.
- Os algoritmos de busca são utilizados em muitas áreas da ciência da computação e são usados para encontrar soluções para problemas complexos em que é necessário procurar em uma grande quantidade de dados para encontrar a melhor resposta possível. Esses algoritmos podem ser divididos em duas categorias principais: busca não informada e busca informada.
    - A busca não informada é um tipo de algoritmo de busca que não possui informações adicionais sobre o problema, além dos dados brutos.
    - A busca informada, também conhecida como busca heurística, usa informações adicionais sobre o problema para encontrar a solução mais rapidamente.

Resolução de problemas utilizando algoritmos de busca

- Além dos algoritmos de busca não informada e informada, existem outros tipos de algoritmos que podem ser usados para resolver problemas, como algoritmos genéticos, algoritmos de otimização e algoritmos de aprendizado de máquina.
- Os **algoritmos genéticos** são usados para resolver problemas de otimização em que é necessário encontrar a melhor solução dentro de um conjunto de possíveis soluções. Esses algoritmos são baseados na seleção natural e na teoria da evolução, em que soluções melhores são selecionadas e combinadas para gerar soluções ainda melhores.
- Os **algoritmos de otimização** são usados para encontrar a melhor solução em um conjunto de possíveis soluções. Eles geralmente usam técnicas matemáticas para encontrar a solução mais próxima possível da solução ideal.
- Os algoritmos de aprendizado de máquina são usados para treinar um modelo com dados de entrada e saída conhecidos, de modo que ele possa prever a saída para novos dados de entrada. Eles são usados em áreas como reconhecimento de voz, detecção de *spam* e análise de sentimentos.

Problemas padronizados

- Os problemas padronizados são problemas comuns que são amplamente utilizados para avaliar o desempenho de sistemas de inteligência artificial. Eles geralmente incluem tarefas específicas, como reconhecimento de imagem, processamento de linguagem natural e jogos de tabuleiro, e são projetados para testar as habilidades de uma IA em áreas específicas. Os resultados desses problemas são comparáveis entre diferentes sistemas, permitindo que os pesquisadores comparem o desempenho de diferentes abordagens e algoritmos.
- Exemplo: o mundo do aspirador de pó.

Problemas do mundo real

- Os problemas do mundo real são problemas que encontramos em nosso ambiente cotidiano e que geralmente são mais complexos e desafiadores do que os problemas padronizados. Eles podem incluir tarefas como conduzir um carro, reconhecimento de fala em ambientes ruidosos, planejamento de rotas em um mapa ou tomar decisões em situações incertas. Esses problemas geralmente requerem uma combinação de habilidades, como percepção, raciocínio e aprendizado, para serem resolvidos e são mais difíceis de serem modelados e testados do que os problemas padronizados. Exemplo:
- O problema do caixeiro viajante que é um problema clássico de otimização combinatória em que se busca encontrar o menor caminho possível que um caixeiro viajante pode percorrer para visitar todas as cidades em um conjunto dado, retornando à cidade de origem.
    - O problema é um dos mais conhecidos da teoria dos grafos e tem aplicações em diversas áreas, como logística, transporte, planejamento urbano, dentre outras.
    - O desafio do problema é encontrar a rota mais curta possível, considerando todas as cidades do conjunto e retornando à cidade de origem, passando por cada cidade uma única vez.

Espaço de estados

- O espaço de estados é um conceito fundamental na inteligência artificial (IA) que se refere ao conjunto de estados possíveis e suas transições para um determinado problema. Ele é usado para modelar problemas de busca e planejamento, como jogos, robótica e inteligência artificial para jogos.
- O espaço de estados é composto por um conjunto de estados, cada um representando uma configuração possível do problema. Esses estados são conectados por ações ou transições, que representam as ações que podem ser tomadas para mudar de um estado para outro. As ações são regidas por regras e restrições, que determinam quais ações são possíveis em cada estado.
    - O espaço de estados é geralmente representado como um grafo, em que cada nó representa um estado e as arestas representam as transições entre os estados. Isso permite que os algoritmos de busca sejam aplicados para encontrar soluções para o problema.
    - O espaço de estados também é usado para planejamento, que é o processo de encontrar uma sequência de ações que leve de um estado inicial para um estado final desejado.

Representação

- A representação do espaço de estados é uma etapa fundamental. A ideia é representar o problema em questão em termos de estados, em que cada estado representa uma configuração ou situação possível do problema. A partir dessa representação, os algoritmos de busca podem explorar o espaço de estados em busca de soluções para o problema.

A representação do espaço de estados pode ser feita de diferentes maneiras, dependendo do tipo de problema em questão. Algumas das formas mais comuns de representação são:

1. Vetor ou matriz de estado.
2. Grafo de estados.
3. Conjunto de regras ou axiomas.
4. Autômatos e máquinas de estado.
5. Árvores de decisão.

Desempenho da resolução de problemas

- Análise da complexidade é uma técnica utilizada para medir o desempenho de algoritmos em inteligência artificial (IA). Ela permite avaliar o tempo e o espaço necessários para que um algoritmo execute e forneça uma medida da eficiência do algoritmo. Isso é importante para garantir que os algoritmos possam ser executados de forma rápida e eficiente, especialmente quando lidamos com grandes conjuntos de dados.
- Existem várias formas de medir a complexidade de um algoritmo, mas a mais comum é a notação O. A notação O fornece uma estimativa do número de operações que um algoritmo realiza em relação ao tamanho do conjunto de dados de entrada.
    - Por exemplo, um algoritmo com complexidade O(n) realiza uma quantidade de operações proporcional ao tamanho do conjunto de dados. Um algoritmo com complexidade O(1) realiza uma quantidade constante de operações independentemente do tamanho do conjunto de dados.
    - A complexidade de um algoritmo depende de vários fatores, incluindo a estrutura de dados utilizada, a implementação do algoritmo e até mesmo a linguagem de programação usada.

Algoritmos de busca – busca cega

- O algoritmo de busca cega (também chamado de busca sem informação) é um método comum utilizado na inteligência artificial para encontrar soluções para problemas de busca. Esses algoritmos são baseados em visitar todos os estados possíveis de um problema, geralmente usando uma estratégia de busca exaustiva.
- A busca em largura é um algoritmo de busca cega que expande os nós de um grafo de busca em largura (como uma árvore) a partir de um nó inicial e visita todos os nós vizinhos antes de se mover para o próximo nível. Isso garante que todos os estados a uma distância específica do nó inicial sejam visitados antes de se mover para estados mais distantes. Isso é útil para problemas em que é desejável encontrar a solução com o menor número de passos.
    - A busca em profundidade é outro algoritmo de busca cega que expande os nós a partir do nó inicial e visita cada nó o mais profundamente possível antes de voltar e explorar outros caminhos. Garante que todos os estados que são “filhos” de um estado específico sejam visitados antes de se mover para outros estados. É útil para problemas em que é desejável encontrar a solução com o menor número de nós.

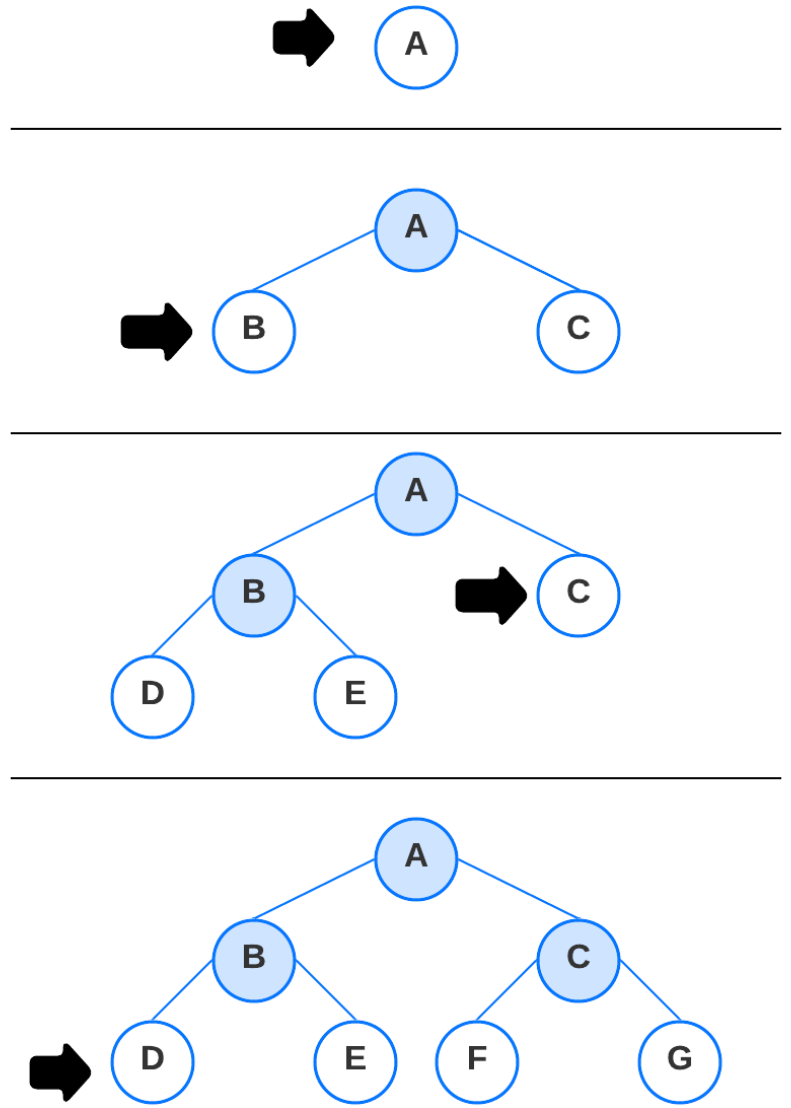
Busca em largura

- A busca em largura é geralmente usada em problemas de busca, em que existe um estado inicial e um estado final, e o objetivo é encontrar o caminho mais curto entre os dois estados. Ela é útil quando não há uma heurística para avaliar a qualidade de um estado, ou quando não é possível prever o estado final.
- A busca em largura é implementada usando uma fila para armazenar os nós visitados. O algoritmo começa explorando todos os nós adjacentes ao estado inicial e adicionando-os à fila. Em seguida, o algoritmo remove o primeiro nó da fila e explora seus nós adjacentes, adicionando-os à fila. Isso é repetido até que o estado final seja encontrado ou até que todos os nós tenham sido explorados.
    - Existem algumas limitações na busca em largura. Uma delas é que ela pode ser ineficiente em problemas de busca em que há muitos estados possíveis, pois ela explorará todos os estados, mesmo se a maioria deles não for relevante para a solução do problema. Além disso, ela também pode ser ineficiente em problemas de busca com muitos nós profundos, pois ela precisará explorar todos os nós em cada profundidade antes de se aprofundar.

Busca em largura

```
função BUSCA-EM-LARGURA(problema) retorna uma solução ou falha
  nó ← um nó com ESTADO = problema.ESTADO-INICIAL, CUSTO-DE-CAMINHO = 0
  se problema.TESTE-DE-OBJETIVO(nó.ESTADO) senão retorne SOLUÇÃO(nó),
  borda ← uma fila FIFO com nó como elemento único
  explorado ← conjunto vazio
  repita
    se VAZIO?(borda), então retorne falha
    nó ← POP(borda) / * escolhe o nó mais raso na borda */
    adicione nó.ESTADO para explorado
    para cada ação em problema.AÇÕES(nó.ESTADO) faça
      filho ← NÓ-FILHO(problema, nó, ação),
      se (filho.ESTADO)não está em explorado ou borda então
        se problema.TESTE-DE-OBJETIVO(filho.ESTADO) então retorne SOLUÇÃO(filho)
        borda ← INSIRA(filho, borda)
```

Fonte: Norvig (2013, p. 70).



Fonte: autoria própria.

Busca em profundidade

- A busca em profundidade é uma técnica de inteligência artificial que é usada para resolver problemas de busca em grafos e árvores. O objetivo é percorrer todos os nós de um grafo ou árvore, explorando completamente cada ramo antes de passar para o próximo. É uma técnica recursiva que segue uma pré-ordem e é frequentemente comparada com a busca em largura, que busca por meio de todos os nós de forma sistemática e uniforme.
- A busca em profundidade começa no nó inicial e adiciona todos os nós adjacentes à pilha. Ele então continua visitando cada nó na pilha até encontrar a solução desejada ou determinar que não há solução.
    - A busca em profundidade tem algumas vantagens sobre a busca em largura. Por exemplo, ela pode ser mais eficiente em espaços de busca maiores e pode encontrar soluções mais curtas em problemas de caminhos mínimos. No entanto, essa técnica também tem algumas desvantagens, como a possibilidade de entrar em *looping* infinito e não garantir que todos os nós sejam visitados.

Profundidade limitada

- A busca em profundidade limitada é uma variação da técnica de busca em profundidade em inteligência artificial que adiciona uma profundidade máxima para evitar *looping* infinito. A busca em profundidade tradicional tem como objetivo percorrer todos os nós de um grafo ou árvore, explorando completamente cada ramo antes de passar para o próximo, mas essa técnica pode entrar em *looping* infinito se não houver um mecanismo para evitar isso. A busca em profundidade limitada adiciona esse mecanismo, limitando a profundidade máxima da busca.
- A busca em profundidade limitada começa no nó inicial e adiciona todos os nós adjacentes à pilha. Ela então continua visitando cada nó na pilha até encontrar a solução desejada ou atingir a profundidade máxima estabelecida. Se a solução não for encontrada, o algoritmo retrocede e tenta novamente a partir de um nó anterior, até que a solução seja encontrada ou todas as possibilidades sejam esgotadas. Ela é menos propensa a entrar em *looping* infinito do que a busca em profundidade tradicional, mas pode não garantir que se encontre a solução ótima, pois pode haver soluções melhores além da profundidade estabelecida.

Busca em profundidade e profundidade limitada

função BUSCA-EM-PROFUNDIDADE-LIMITADA(problema, limite) **retorna** uma solução ou falha/corte

retornar BPL-RECURSIVA (CRIAR-NÓ(problema, ESTADO-INICIAL), problema, limite)

função BPL-RECURSIVA(nó, problema, limite) **retorna** uma solução ou falha/corte

  se problema. TESTAR-OBJETIVO (nó.ESTADO) **então, retorna** SOLUÇÃO (nó)

  se não se limite = 0 **então retorna** corte

  senão

    corte_ocorreu? ← falso para cada ação no problema.AÇÕES(nó.ESTADO) **faça**

      filho ← NÓ-FILHO (problema, nó, ação)

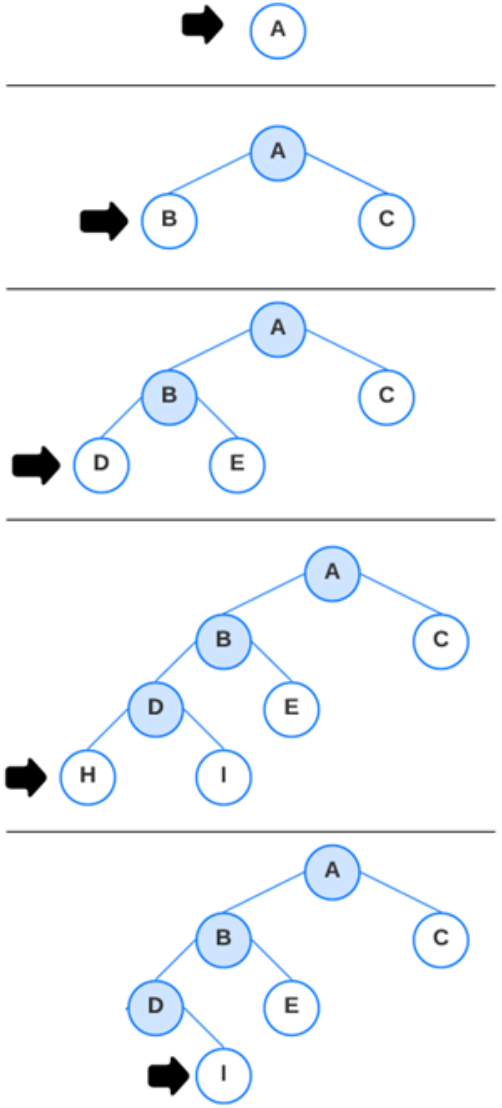
      resultado ← BPL-RECURSIVA (criança, problema limite – 1)

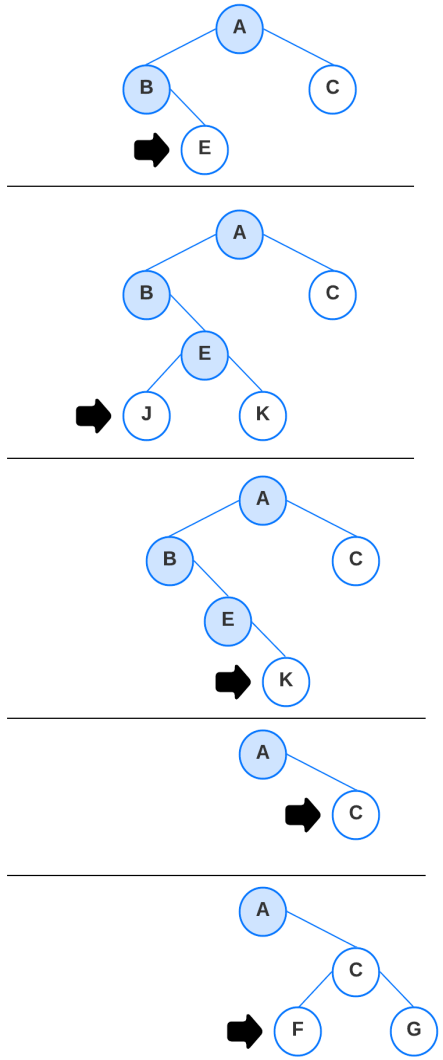
      se resultado = corte **então** corte_ocorreu? ← verdadeiro

      senão se resultado ≠ falha **então retorna** resultado

    se corte_ocorreu? **então retorna** corte **senão retorna** falha

Fonte: NORVIG (2013, p. 74)





Fonte: autoria própria.

Busca em profundidade iterativa

- O algoritmo de busca em profundidade iterativa (ou IDDFS, do inglês *Iterative Deepening Depth-First Search*) é uma estratégia de busca em grafos que combina as vantagens da busca em profundidade com a eficiência da busca em largura.
- A ideia básica do algoritmo é executar sucessivas buscas em profundidade, limitando a profundidade máxima em cada busca, até encontrar o nó desejado. A profundidade máxima é aumentada a cada iteração e o algoritmo para assim que encontra o nó desejado ou quando alcança a profundidade máxima estabelecida.

O IDDFS funciona da seguinte maneira:

1. Inicia-se a busca a partir do nó raiz com profundidade máxima 0.
2. Se o nó atual é o nó de destino, a busca termina e o caminho é retornado.
3. Se a profundidade máxima foi atingida e o nó de destino não foi encontrado, retorna-se para o nó anterior.
4. Caso contrário, expande-se o nó atual e adicionam-se seus filhos à fronteira.
5. Repete-se o processo para o próximo nó na fronteira.

Busca em profundidade iterativa

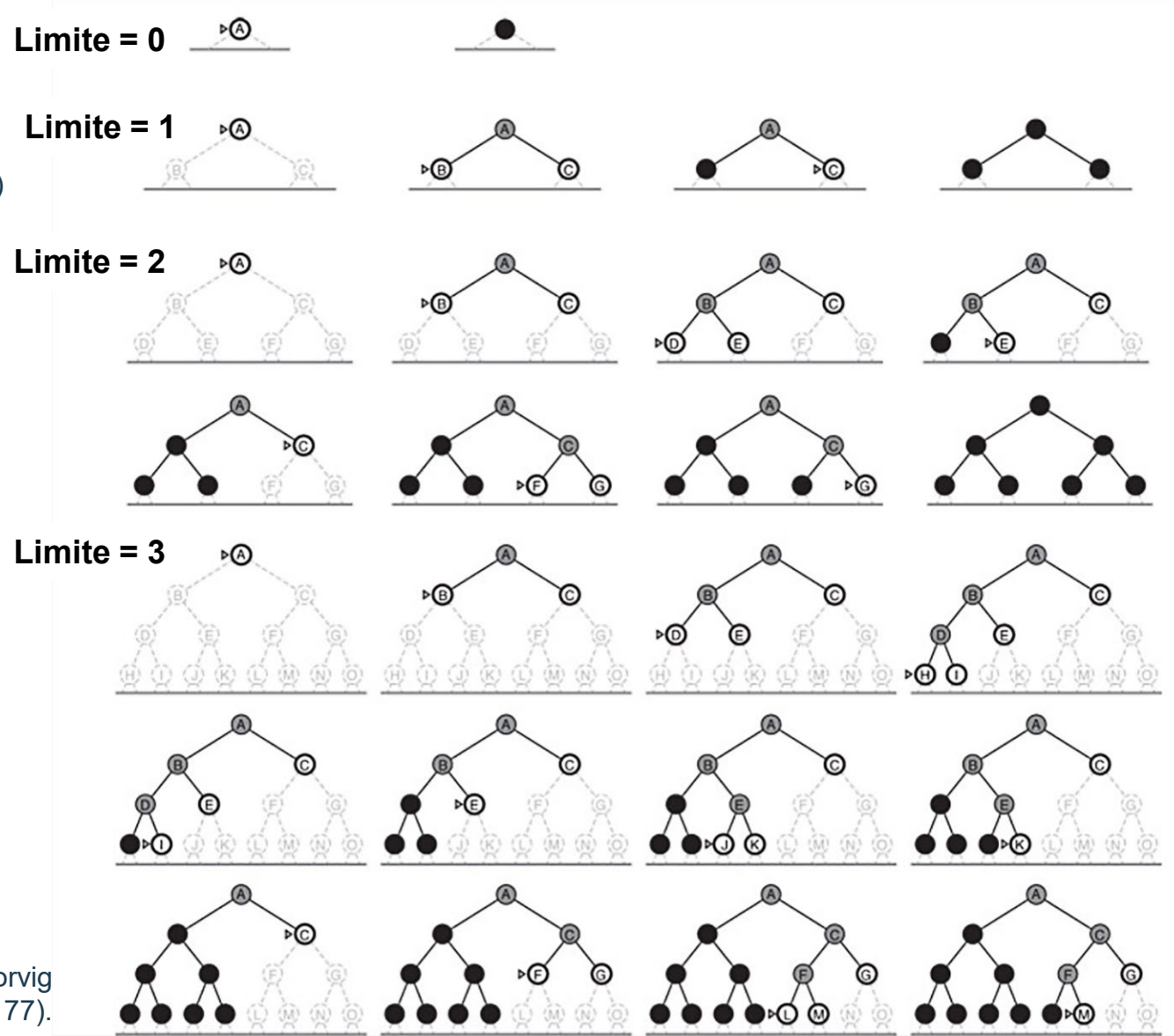
função BUSCA-DE-APROFUNDAMENTO-ITERATIVO(problema) **retorna** uma solução ou falha

para profundidade = 0 **até** ∞ **faça**

resultado ← BUSCA-EM-PROFUNDIDADE-LIMITADA(problema, profundidade)

se resultado ≠ corte **então** retornar resultado

Fonte: NORVIG (2013, p. 75)



Fonte: adaptado de: Norvig (2013, p. 77).

Interatividade

Analise as afirmações:

I. Os algoritmos de busca podem ser divididos em duas categorias principais: busca não informada e busca informada.
II. A busca em largura é implementada usando uma fila para armazenar os nós visitados.
III. A busca em profundidade começa no nó inicial e adiciona todos os nós adjacentes à fila.
IV. A busca em profundidade limitada começa no nó inicial e adiciona todos os nós adjacentes à fila.
V. O algoritmo de busca em profundidade iterativa é uma estratégia de busca em grafos que combina as vantagens da busca em profundidade com a eficiência da busca em largura.

Estão corretas:

a) I, II e III.
b) I, II e V.
c) II, III e V.
d) III, IV e V.
e) Todas as afirmações.

Resposta

Analise as afirmações:

I. Os algoritmos de busca podem ser divididos em duas categorias principais: busca não informada e busca informada.
II. A busca em largura é implementada usando uma fila para armazenar os nós visitados.
III. A busca em profundidade começa no nó inicial e adiciona todos os nós adjacentes à fila.
IV. A busca em profundidade limitada começa no nó inicial e adiciona todos os nós adjacentes à fila.
V. O algoritmo de busca em profundidade iterativa é uma estratégia de busca em grafos que combina as vantagens da busca em profundidade com a eficiência da busca em largura.

Estão corretas:

a) I, II e III.
b) I, II e V.
c) II, III e V.
d) III, IV e V.
e) Todas as afirmações.

Algoritmos de busca informada

- Os algoritmos de busca informada (também conhecidos como algoritmos de busca heurística) são uma classe de algoritmos de busca em grafos que utilizam informações adicionais para guiar a busca em direção ao objetivo. Essas informações podem ser heurísticas, ou seja, estimativas do quão perto um nó está do objetivo, ou informações sobre a estrutura do grafo que está sendo pesquisado.

- Os algoritmos de busca informada são úteis em problemas em que o grafo é grande e a busca em profundidade ou busca em largura não são eficientes o suficiente. Esses algoritmos são capazes de guiar a busca para as áreas mais promissoras do grafo, reduzindo o número de nós visitados e, portanto, melhorando a eficiência da busca.

Heurísticas

- As heurísticas são funções que fornecem uma estimativa do custo de chegar ao objetivo a partir de um estado dado. Elas são utilizadas na inteligência artificial para guiar a busca em problemas com espaços de estado grandes e complexos, como planejamento, roteamento, alocação de recursos e outras tarefas que envolvem a busca de uma solução ótima ou subótima.

- As heurísticas são geralmente específicas para cada problema e podem ser construídas com base em conhecimento humano sobre o problema ou aprendidas com dados. Exemplo de heurísticas incluem a distância Manhattan para problemas de roteamento em um *grid*.

    - A heurística de distância Manhattan é uma técnica utilizada em algoritmos de busca informada, para estimar o custo de um caminho de um nó até o objetivo em um grafo. Essa heurística leva em conta apenas as posições dos nós no grafo, ignorando a existência de obstáculos ou outras informações adicionais.

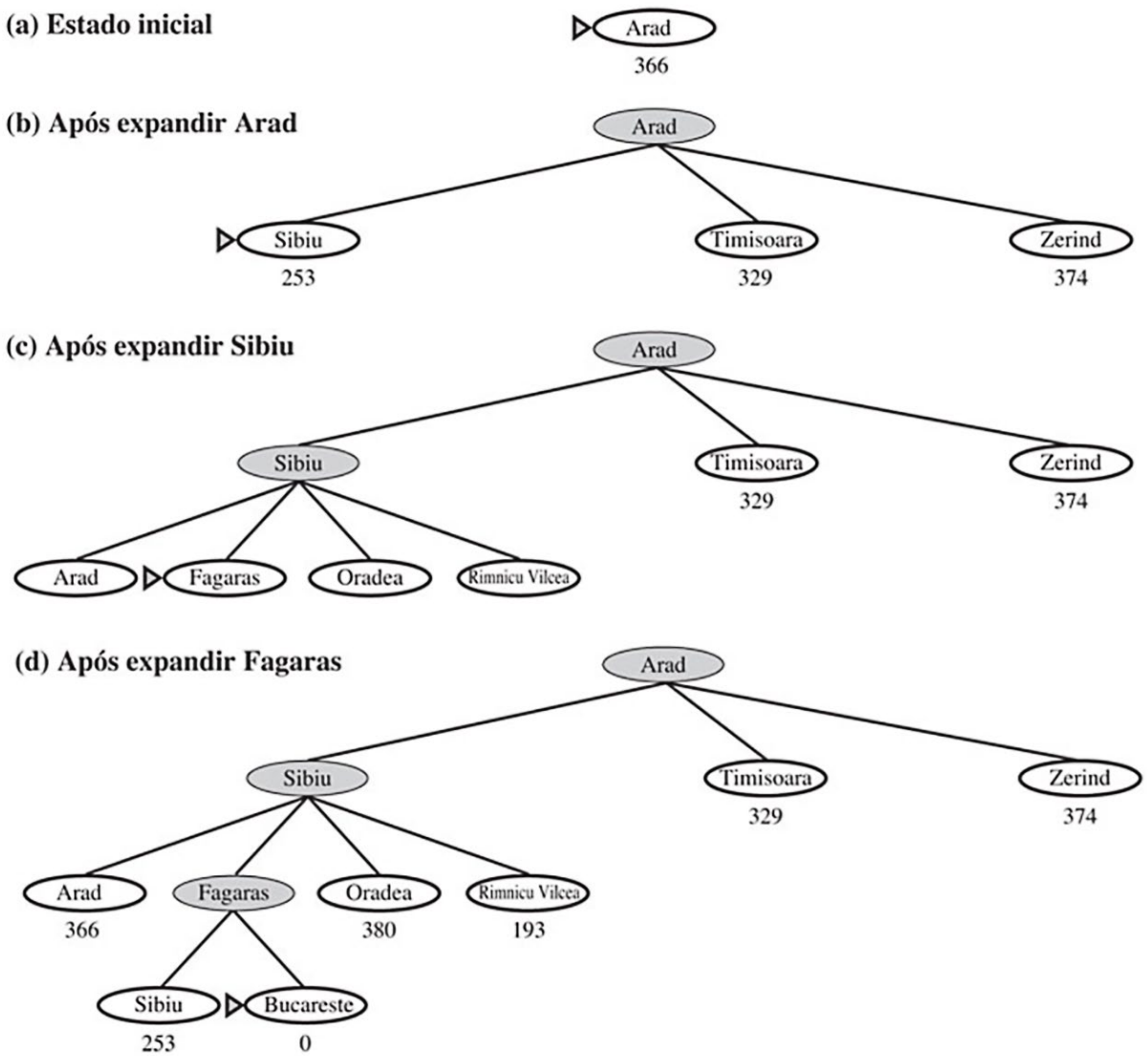
Busca gulosa

- A busca gulosa é um algoritmo de busca heurística que é usado para encontrar soluções aproximadas em problemas de otimização em que o objetivo é encontrar a melhor solução possível entre muitas possíveis soluções. Nesse algoritmo, a cada passo, é feita a escolha que parece ser a melhor no momento, sem levar em consideração as consequências futuras dessa escolha.
- Na busca gulosa, o algoritmo avalia todos os possíveis caminhos a partir do estado atual e escolhe o caminho que parece mais promissor, com base em uma heurística específica. A heurística é usada para estimar a distância ou o custo do caminho para o objetivo final e é essa estimativa que é usada para determinar qual caminho seguir.
    - O objetivo da busca gulosa é encontrar uma solução rapidamente, mesmo que ela não seja necessariamente a melhor solução possível. Em outras palavras, a busca gulosa é um algoritmo que busca uma solução que seja “suficientemente boa”, mas não necessariamente ótima.

Busca gulosa

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

Fonte: Adaptado de: NORVIG (2013, p. 79)



Fonte: Norvig (2013, p. 80).

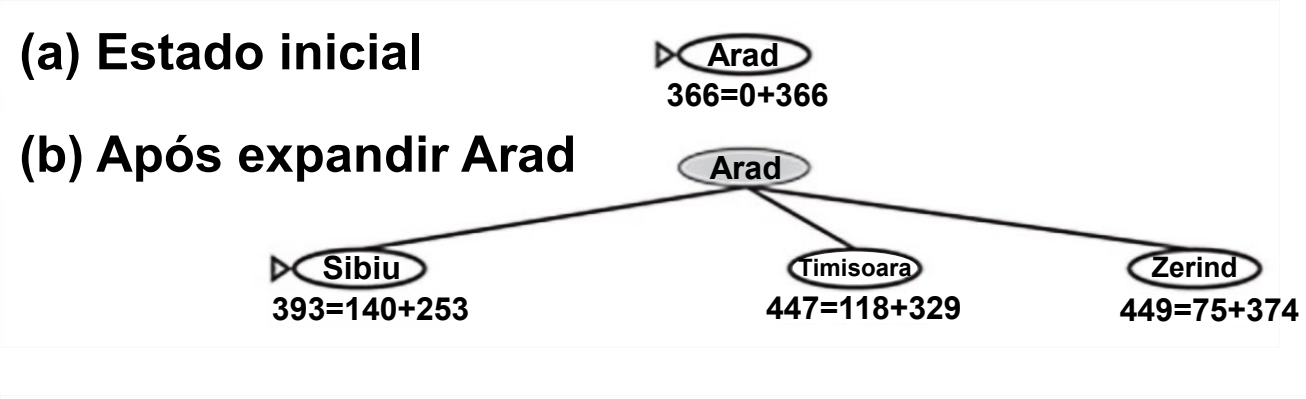
Busca A*

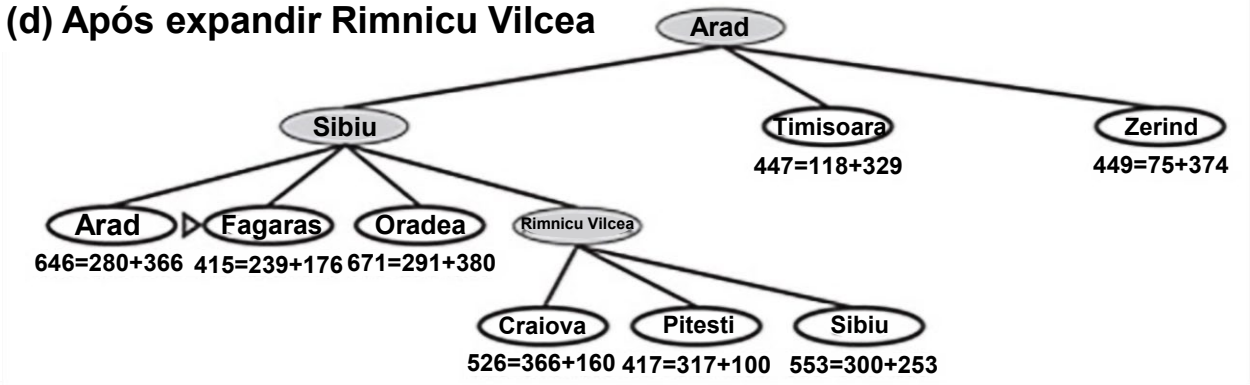
- A Busca A* é um algoritmo de busca informada que utiliza uma heurística, uma função que estima o custo de chegar ao objetivo a partir de um estado dado, para guiar a busca. Ela é utilizada em tarefas de planejamento, roteamento, alocação de recursos e outras tarefas que envolvem a busca de uma solução ótima.
- A* funciona expandindo os nós do espaço de estado de acordo com o valor da função f, que é a soma do custo g (custo até o nó) e do custo h (estimativa do custo até o objetivo). O nó com o menor valor de f é selecionado para ser expandido.
- A* utiliza uma estrutura de dados, como uma fila de prioridade, para armazenar os nós expandidos e ordená-los de acordo com o valor de f. Assim, sempre que um novo nó é expandido, ele é adicionado à fila de prioridade e o nó com o menor valor de f é selecionado para ser expandido.

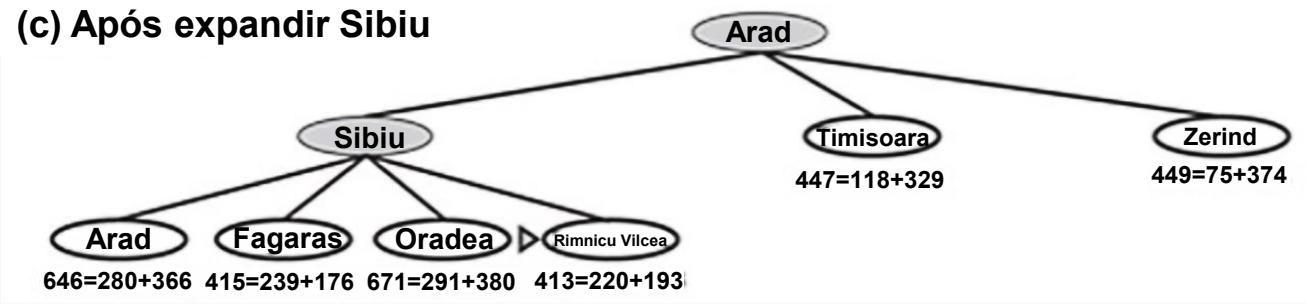
f(n) = custo estimado da solução de menor custo por meio de n.

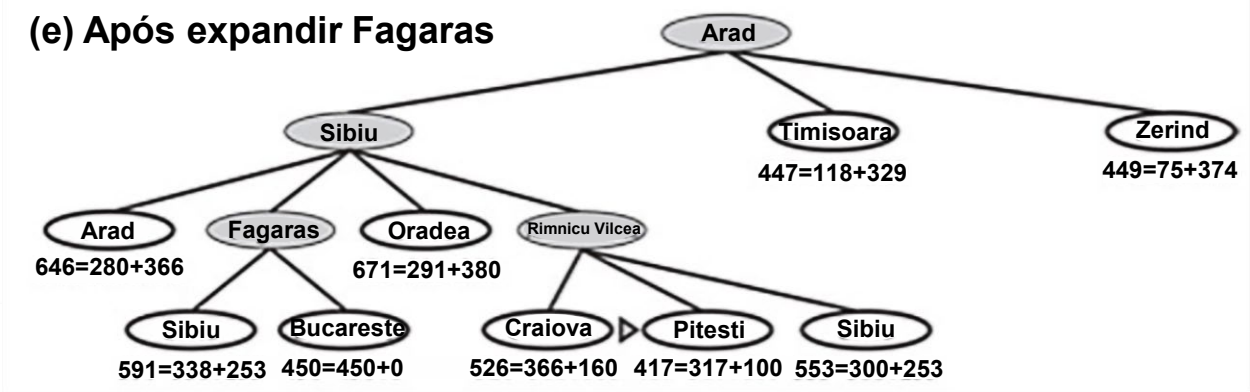
f(n) = g(n) + h(n).

Busca A*





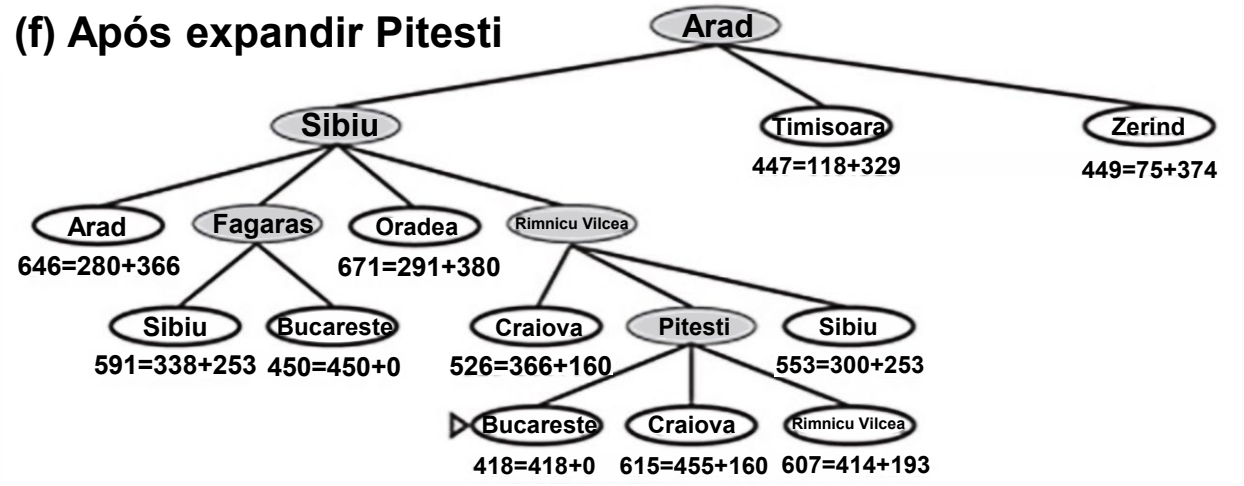




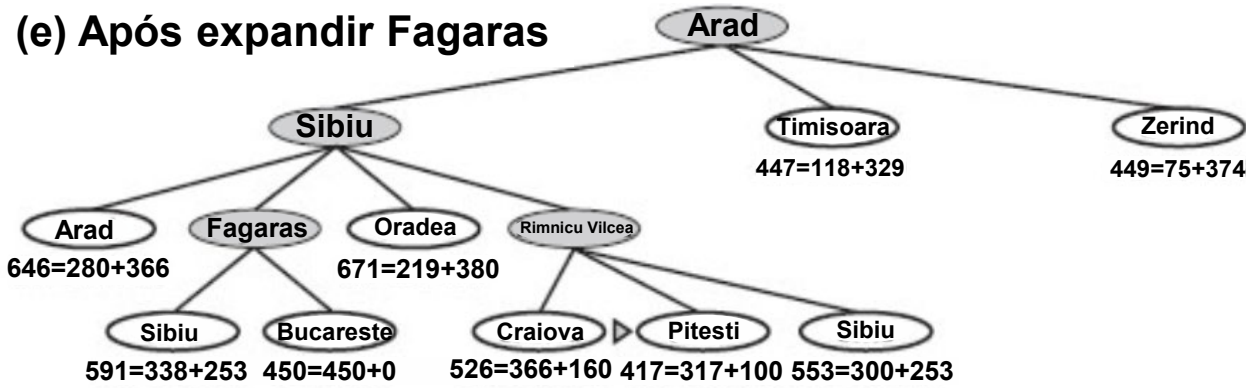
Fonte: adaptado de: Norvig (2013, p. 81).

Busca A*

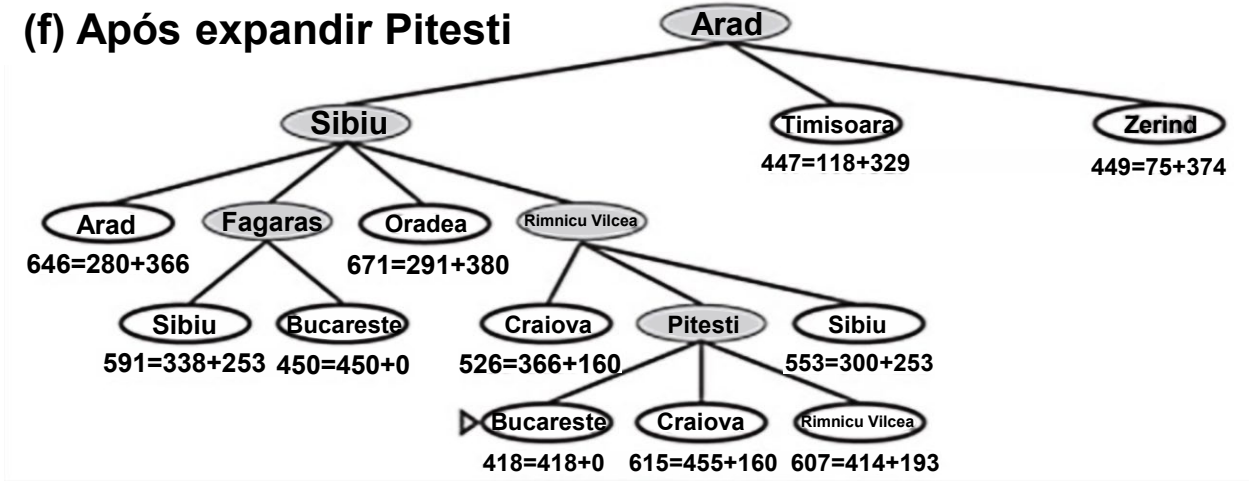
(f) Após expandir Pitesti



(e) Após expandir Fagaras



(f) Após expandir Pitesti



Fonte: adaptado de: Norvig (2013, p. 81).

Busca Best-First

- O algoritmo Best-First é um algoritmo de busca heurística usado para encontrar caminhos em grafos. Ele é um tipo de algoritmo guloso, pois escolhe o próximo nó para explorar com base na heurística que indica qual nó parece ser o “melhor” em direção ao objetivo.

- O algoritmo Best-First começa explorando o nó inicial e, em seguida, escolhe o próximo nó a ser explorado com base na heurística escolhida. Essa heurística pode ser uma estimativa da distância do nó atual até o objetivo, por exemplo. O algoritmo escolhe o nó que parece estar mais próximo do objetivo, de acordo com a heurística, e continua a explorar os nós adjacentes a ele.

    - O algoritmo Best-First não considera o custo total do caminho percorrido até o nó atual, mas apenas a heurística escolhida. Isso pode levar a soluções subótimas em alguns casos, pois o algoritmo pode se concentrar em um caminho que parece promissor de acordo com a heurística, mas que pode ser mais caro do que outro caminho mais longo que leva a uma solução melhor.

Busca IDA*

- A Busca IDA* (*Iterative Deepening* A*) é um algoritmo de busca informada que combina a busca em profundidade com a busca A*. Ela funciona expandindo os nós do espaço de estado em ordem de profundidade. Em cada iteração, um limite de custo é estabelecido e somente os nós com custo menor que o limite são expandidos. O limite é aumentado a cada iteração até que a solução ótima seja encontrada.

- IDA* utiliza uma heurística, uma função que estima o custo de chegar ao objetivo a partir de um estado dado, para guiar a busca. A heurística é usada para calcular o custo f do nó, que é a soma do custo g (custo até o nó) e do custo h (estimativa do custo até o objetivo). O limite é estabelecido como o menor valor f encontrado entre os nós não expandidos.

    - Uma das vantagens de IDA* é que ele é eficiente para lidar com problemas com espaços de estado muito grandes, pois ele não precisa armazenar todos os nós expandidos em memória, somente os nós que estão na fronteira da busca.

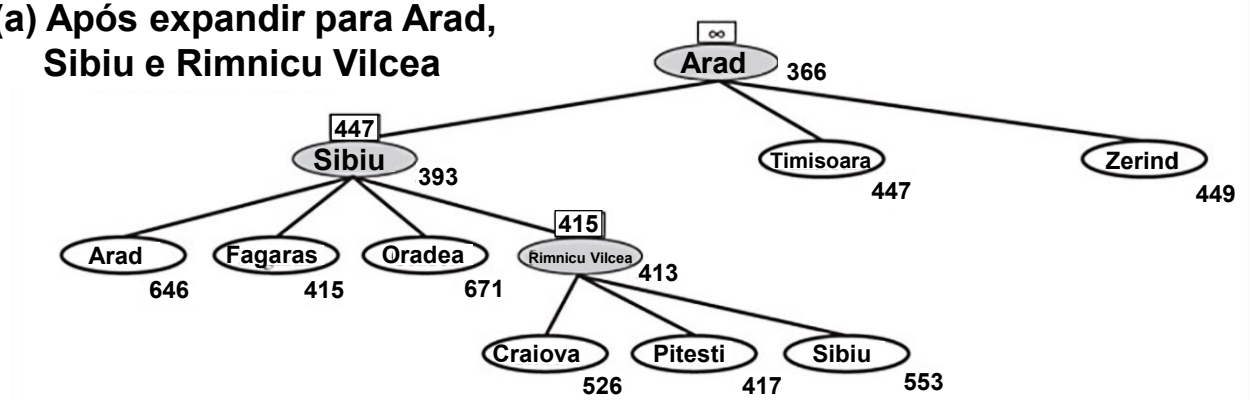
Busca Recursiva Best-first (RBFS)

- A Busca Recursiva Best-first (RBFS) é um algoritmo de busca informada que combina a busca recursiva com a busca best-first. RBFS funciona expandindo os nós do espaço de estado recursivamente e, a cada chamada recursiva, ele seleciona o nó com a melhor avaliação heurística para ser expandido. A diferença entre o RBFS e outros algoritmos de busca best-first é que o RBFS não utiliza uma fila de prioridade para armazenar os nós expandidos. Em vez disso, ele utiliza a chamada recursiva para organizar os nós.

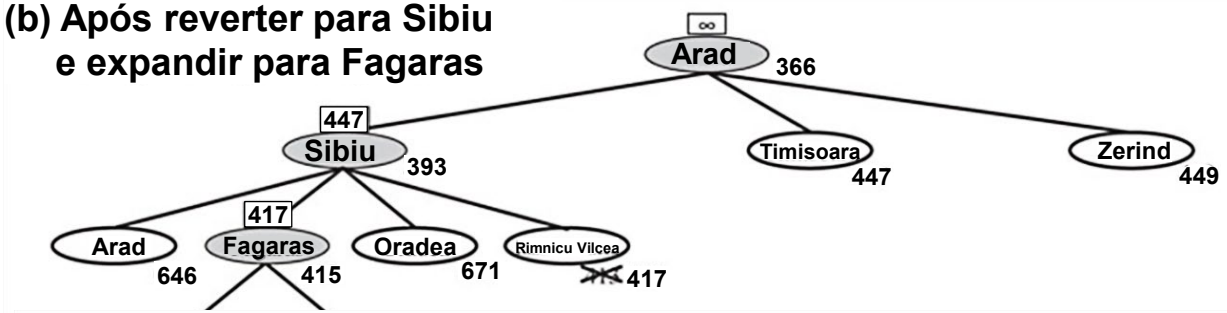
Busca Recursiva Best-first (RBFS)

```
função BUSCA-RECURSIVA-PELA-MELHOR(problema) retorna uma solução ou falha
  retorna RBFS(problema, FAZ-NÓ(problema. ESTADO-INICIAL), ∞)
função RBFS (problema, nó, f_limite) retorna uma solução ou falha e um limite novo f_custo
  se problema. TESTE-OBJETIVO(nó.ESTADO) então retorne SOLUÇÃO (nó)
  sucessores ← [ ]
  para cada ação em problema. AÇÕES(nó.ESTADO) fazer
    adicionar NÓ-FILHO(problema, nó, ação) em sucessores
  se sucessores estiver vazio então retornar falha), ∞
  para cada s em sucessores fazer / * atualizar f com o valor da busca anterior, se houver */
    s.f ← max(s.g + s.h, nó.f))
  repita
    melhor ← valor f mais baixo do nó em sucessores
    se melhor.f > f_limite então retornar falha, melhor.f
    alternativa ← segundo valor f mais baixo entre sucessores
    resultado, melhor.f ← RBFS(problema, melhor, min(f_limite, alternativa)
    se resultado ≠ falha então retornar resultado
```

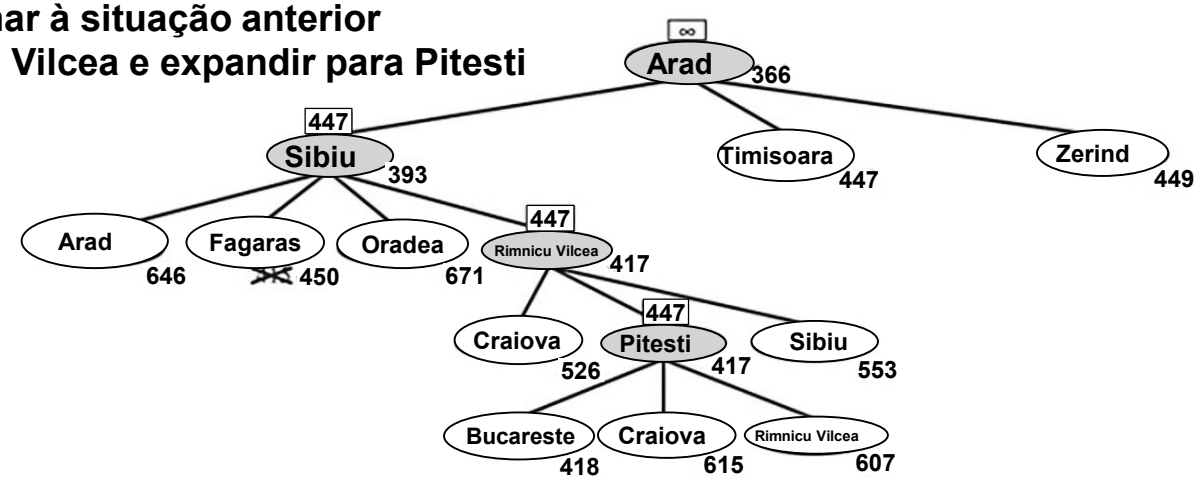
(a) Após expandir para Arad, Sibiu e Rimnicu Vilcea



(b) Após reverter para Sibiu e expandir para Fagaras



(c) Após retornar à situação anterior em Rimnicu Vilcea e expandir para Pitesti



Fonte: adaptado de: Norvig (2013, p. 85).

Busca escalada na montanha (*hill-climbing*)

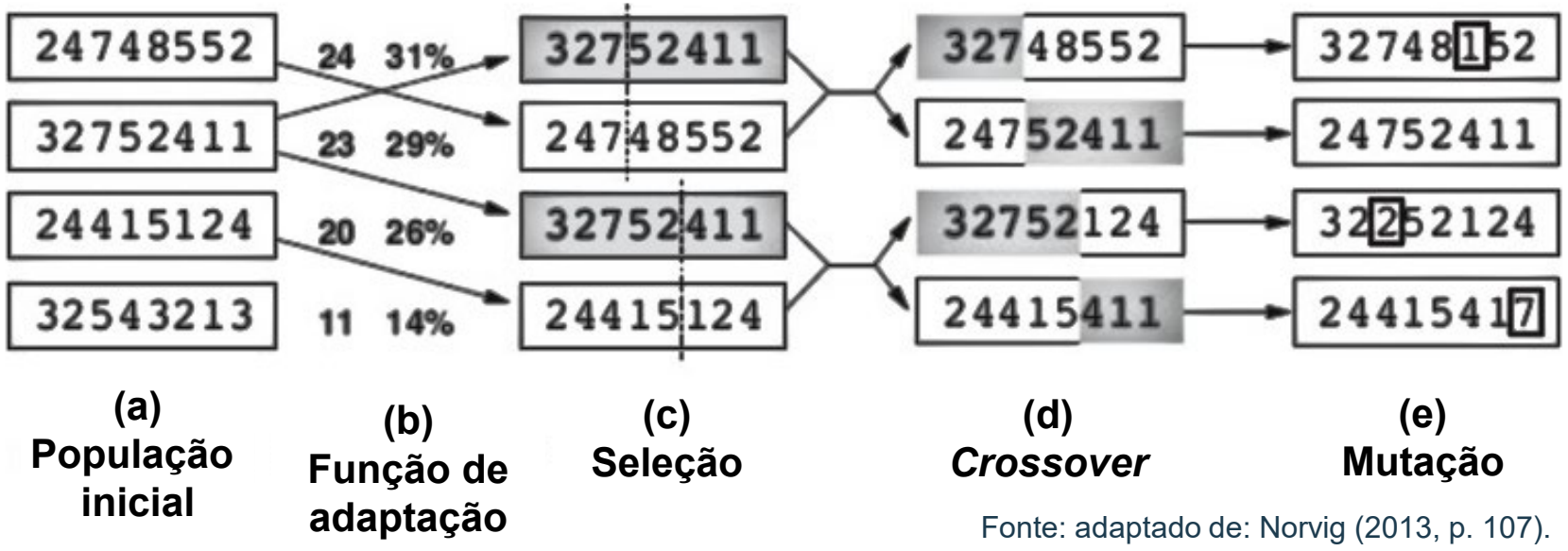
- A busca escalada na montanha (*hill-climbing*) é um algoritmo de busca local utilizado para encontrar a solução ótima ou subótima de um problema. A ideia básica do algoritmo é começar com uma solução inicial e, a cada iteração, gerar sucessores para a solução atual e escolher o sucessor que melhor se aproxima da solução ótima. A busca é interrompida quando uma solução local ótima é encontrada ou quando não há mais melhorias.
- O nome “Escalada na Montanha” é uma analogia ao movimento que um alpinista faria ao subir uma montanha, em que ele sempre tenta subir em direção ao ponto mais alto e não necessariamente precisa saber qual é o ponto mais alto da montanha.
    - O algoritmo de busca Escalada na Montanha é simples de implementar e tem um bom desempenho para problemas pequenos, mas pode se sair mal em problemas maiores ou com muitos mínimos locais, pois o algoritmo pode ficar preso em um mínimo local e não encontrar a solução global. Além disso, se a função de avaliação não for bem escolhida, a busca pode ser ineficiente e nunca encontrar a solução ótima.

Algoritmos genéticos

- Os algoritmos genéticos são uma técnica de busca inspirada na evolução biológica, amplamente utilizada em inteligência artificial e otimização. Eles são utilizados para encontrar soluções ótimas ou subótimas para problemas complexos e de alta dimensionalidade que são difíceis de serem resolvidos com algoritmos convencionais.
- Os algoritmos genéticos funcionam criando uma população de soluções candidatas e, por meio de operações de seleção, cruzamento e mutação, evoluindo essa população ao longo de várias gerações. A cada geração, as soluções são avaliadas usando uma função de *fitness* e as melhores soluções são selecionadas para gerar a próxima geração. O processo é repetido até que uma solução satisfatória seja encontrada ou até que o algoritmo atinja o número máximo de gerações.
    - Os algoritmos genéticos são altamente parametrizáveis, permitindo ajustar coisas como a taxa de cruzamento, a taxa de mutação, o tamanho da população, entre outras coisas, para adaptar o algoritmo para o problema específico.

Algoritmos genéticos

- O algoritmo genético é ilustrado por sequências de dígitos que representam os estados das oito rainhas. A população inicial em (a) é classificada pela função de adaptação em (b), resultando em pares de correspondência em (c). Eles produzem descendentes em (d), sujeitos à mutação em (e).



Fonte: adaptado de: Norvig (2013, p. 107).

Interatividade

Analise as afirmações:

I. A Busca A* é um algoritmo de busca informada que utiliza uma heurística.
II. O algoritmo Best-First utiliza busca heurística e é usado para encontrar caminhos em grafos.
III. A Busca Recursiva Best-first (RBFS) é um algoritmo de busca informada que combina a busca recursiva com a busca best-first.
IV. A busca escalada na montanha (*hill-climbing*) é um algoritmo de busca local utilizado para encontrar a solução ótima ou subótima de um problema.
V. Os algoritmos genéticos são uma técnica de busca inspirada na evolução biológica, amplamente utilizada em inteligência artificial e otimização.

Estão corretas:

a) I, II e III.
b) I, II e IV.
c) I, II e V.
d) III, IV e V.
e) Todas as afirmações.

Resposta

Analise as afirmações:

I. A Busca A* é um algoritmo de busca informada que utiliza uma heurística.
II. O algoritmo Best-First utiliza busca heurística e é usado para encontrar caminhos em grafos.
III. A Busca Recursiva Best-first (RBFS) é um algoritmo de busca informada que combina a busca recursiva com a busca best-first.
IV. A busca escalada na montanha (*hill-climbing*) é um algoritmo de busca local utilizado para encontrar a solução ótima ou subótima de um problema.
V. Os algoritmos genéticos são uma técnica de busca inspirada na evolução biológica, amplamente utilizada em inteligência artificial e otimização.

Estão corretas:

a) I, II e III.
b) I, II e IV.
c) I, II e V.
d) III, IV e V.
e) Todas as afirmações.

Referências

- NORVIG, Peter. *Inteligência Artificial*. 3. ed. São Paulo: Grupo GEN, 2013.

ATÉ A PRÓXIMA!