

# Aprendizado por Reforço

## Prof. Domingos Napolitano

Aula 1: Introdução ao Aprendizado por Reforço

# Aprendizado por Reforço

**Disciplina: IBM8919 - 6º Semestre 2026.1**

**Graduação em Ciência de Dados e  
Inteligência Artificial**

**Prof. Dr. Domingos M R Napolitano**

**Quartas Feiras : 9:40 – 11:40**

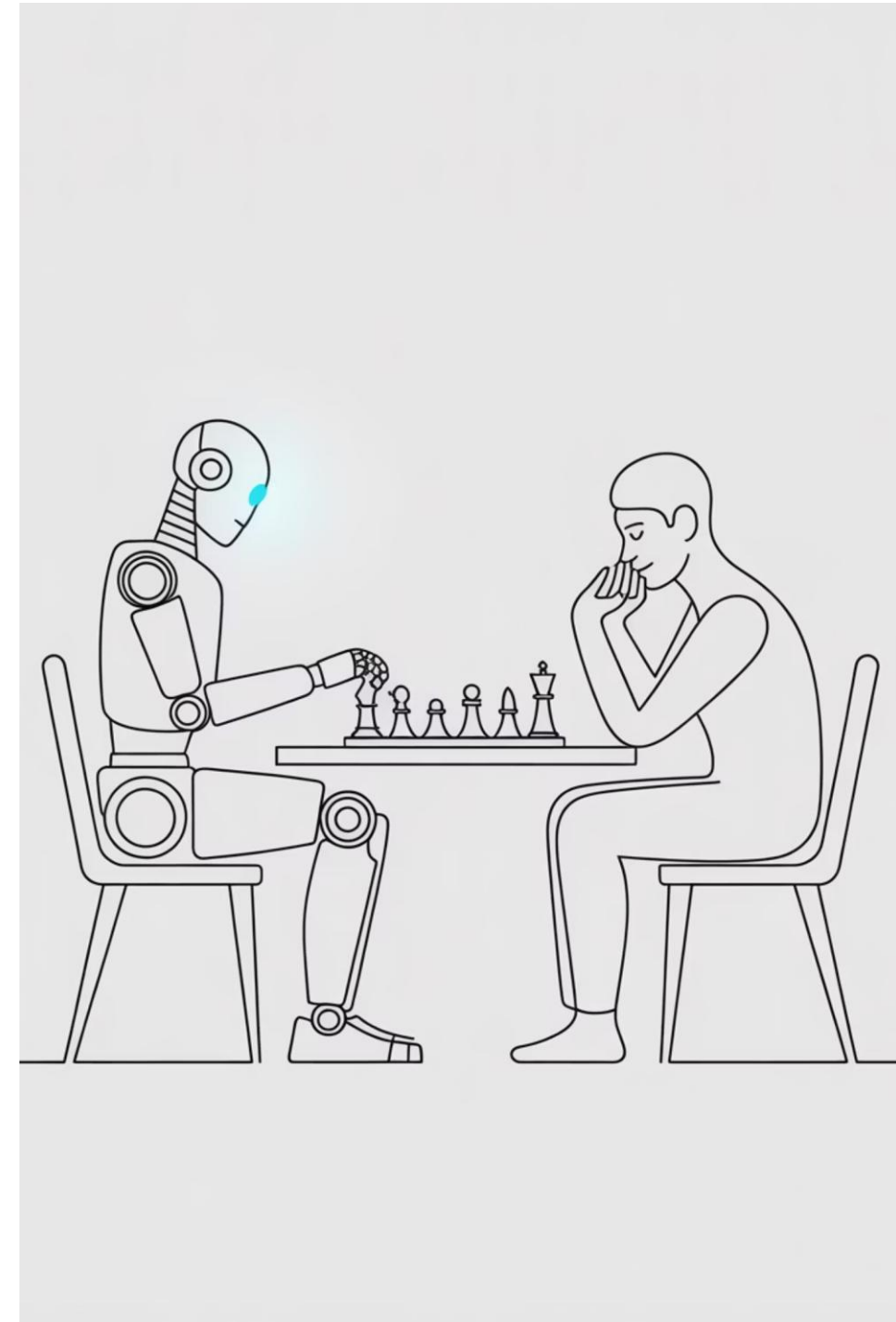
**Sextas Feiras : 7:40 – 9:40 h**

**DOMINGOS.NAPOLITANO@professores.ibmec.edu.br**



# Aprendizado por Reforço: Breve Introdução

Bem-vindos a uma experiência transformadora que vai muito além de um simples curso. Juntos, embarcaremos em uma jornada fascinante pelo universo do aprendizado por reforço, uma das áreas mais revolucionárias da inteligência artificial moderna.



# Agenda do Semestre

**1**

## Fundamentos Sólidos

K-armed bandits e o dilema entre exploração e aproveitament

**2**

## Decisões Sequenciais

Cadeias de Markov e Processos de Decisão de Markov (MDPs)

**3**

## Algoritmos Clássicos

Programação Dinâmica, Value Iteration e Policy Iteration

**4**

## Aprendizado na Incerteza

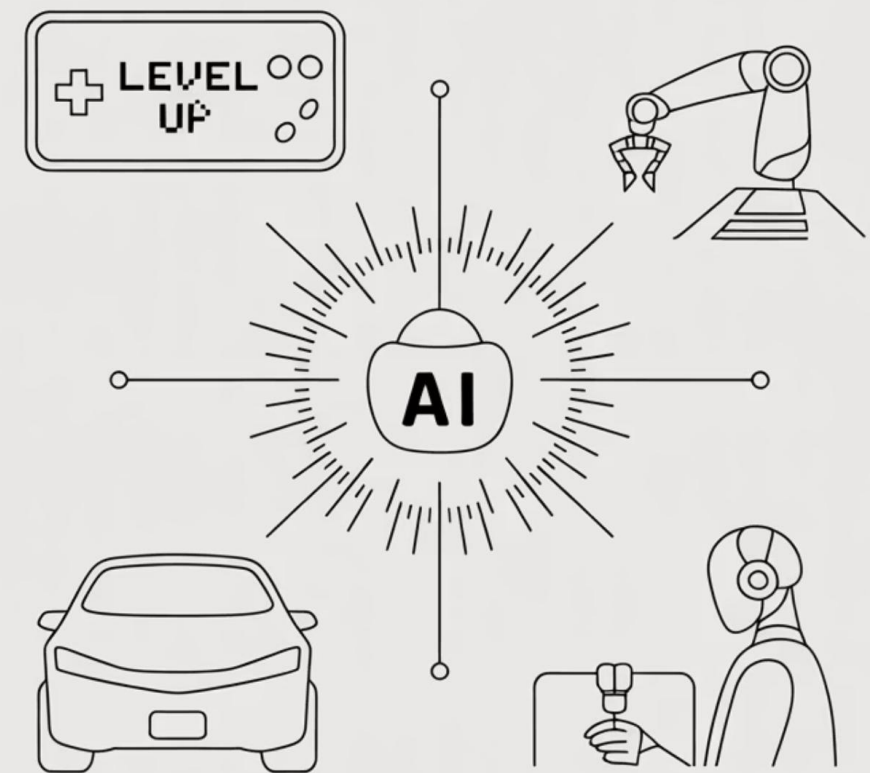
Métodos de Monte Carlo, Q-Learning, SARSA e além

Este curso foi projetado para equilibrar teoria robusta com prática intensiva, permitindo que você domine progressivamente os conceitos e aplicações do aprendizado por reforço. Cada aula construirá sobre o conhecimento anterior, culminando em uma compreensão profunda desta tecnologia transformadora.

# Por Que O Aprendizado por Reforço Está Transformando o Mundo?

O aprendizado por reforço representa uma mudança de paradigma na inteligência artificial, capacitando máquinas a aprenderem através da experiência, assim como humanos. Esta abordagem revolucionária está remodelando indústrias inteiras e criando possibilidades antes inimagináveis.

Diferente de outras formas de aprendizado de máquina, o aprendizado por reforço não depende de exemplos rotulados. Em vez disso, os agentes aprendem através de tentativa e erro, recebendo feedback do ambiente na forma de recompensas. Esta capacidade de aprender fazendo torna o aprendizado por reforço excepcionalmente poderoso para resolver problemas complexos em ambientes dinâmicos.



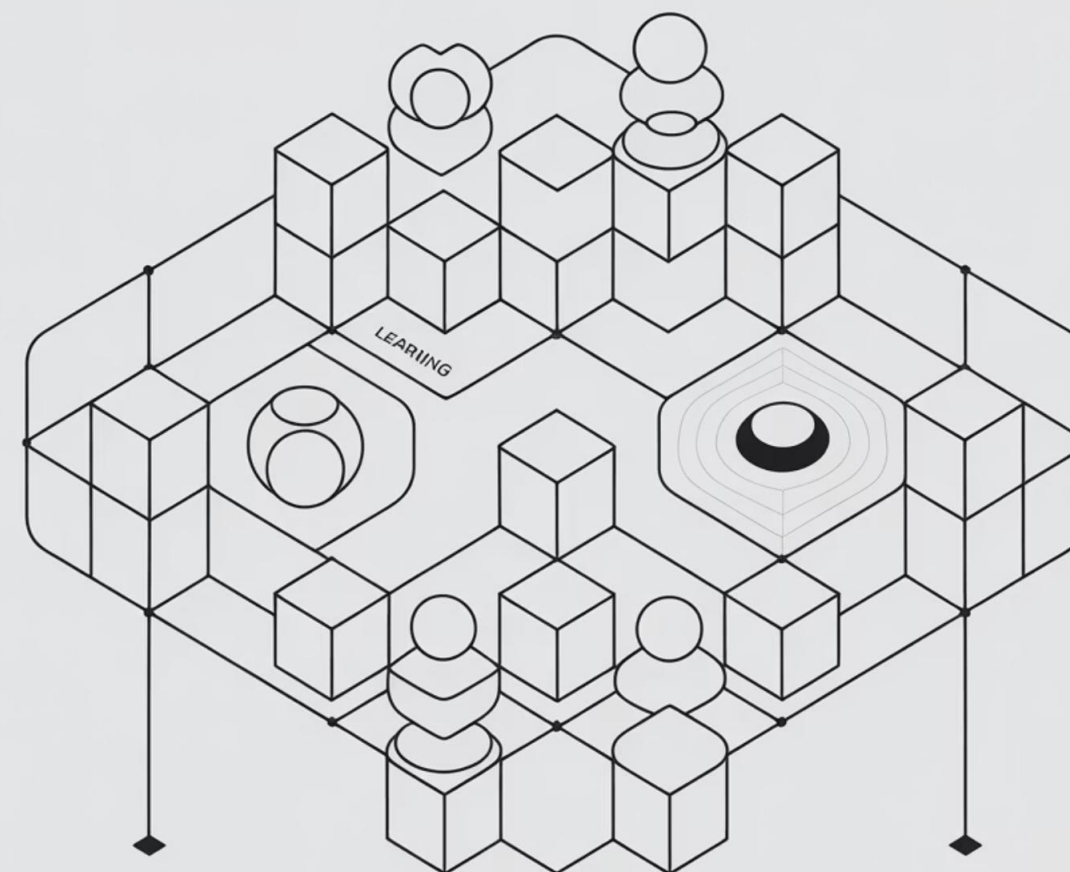
# Revolucionando os Jogos

Os jogos representam um ambiente perfeito para o desenvolvimento e teste de algoritmos de aprendizado por reforço, oferecendo desafios complexos em um ambiente controlado. As conquistas nesta área têm sido verdadeiramente revolucionárias:

- **AlphaGo e AlphaZero:** Derrotaram campeões mundiais em jogos considerados impossíveis para computadores apenas alguns anos atrás
- **OpenAI Five:** Competiu em Dota 2, um dos jogos de estratégia em tempo real mais complexos, exigindo coordenação de equipe e planejamento de longo prazo
- **StarCraft II:** Agentes desenvolveram estratégias inovadoras nunca antes vistas, mesmo por jogadores profissionais com anos de experiência

Estas conquistas não são apenas impressionantes por si só, mas demonstram a capacidade do aprendizado por reforço de desenvolver soluções criativas para problemas extremamente complexos.

O fascinante no caso do AlphaZero é que ele aprendeu jogando contra si mesmo, sem conhecimento humano especializado além das regras básicas. Em apenas 24 horas de auto-aprendizado, conseguiu superar os melhores programas de xadrez tradicionais que haviam sido refinados por décadas.





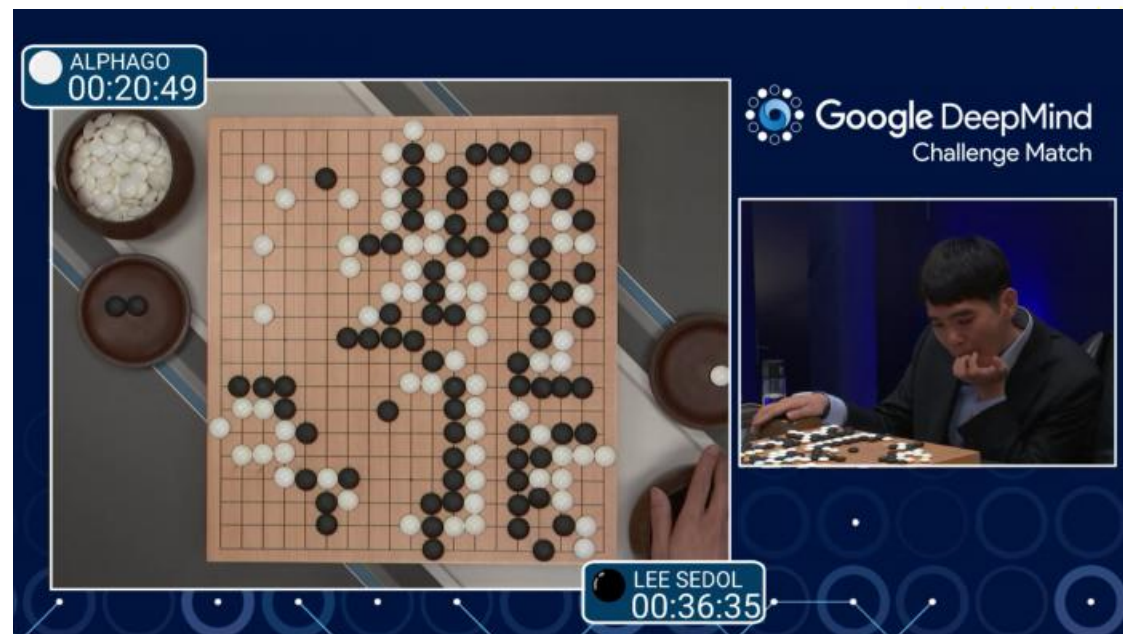
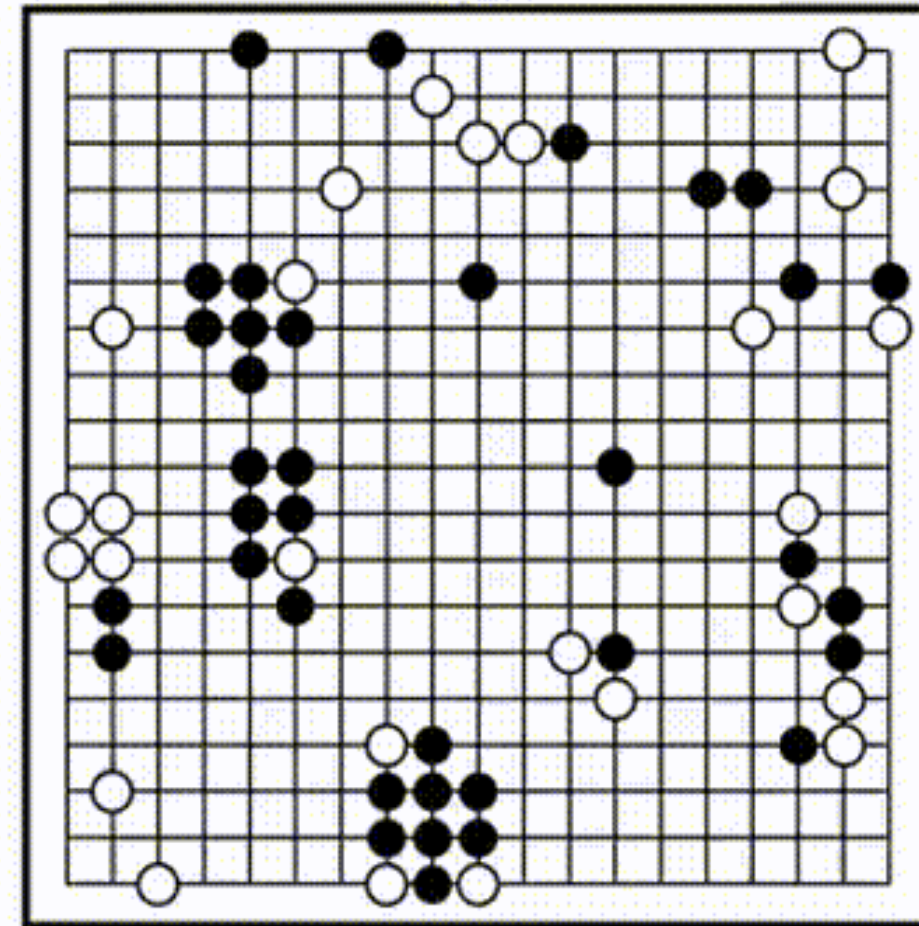
# O que é Aprendizado por Reforço?

O Aprendizado por Reforço (RL) é um paradigma de aprendizado de máquina onde agentes autônomos aprendem a tomar decisões sequenciais em ambientes incertos, através de interações baseadas em recompensas.



O agente desenvolve uma política para maximizar a recompensa cumulativa ao longo do tempo, sem supervisão explícita.

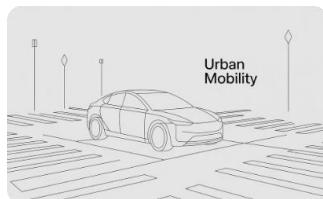
# O AlphaGO



Não perca o documentário AlphaGO no You Tube

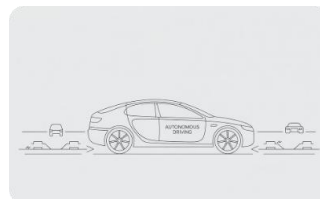


# Condução Autônoma: Navegando um Mundo Imprevisível



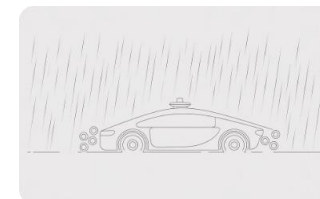
## Navegação em Trânsito Complexo

Os veículos autônomos empregam aprendizado por reforço para desenvolver estratégias robustas de navegação em ambientes urbanos altamente dinâmicos e imprevisíveis. Cada interação com pedestres, ciclistas e outros veículos serve como oportunidade de aprendizado.



## Decisões em Tempo Real

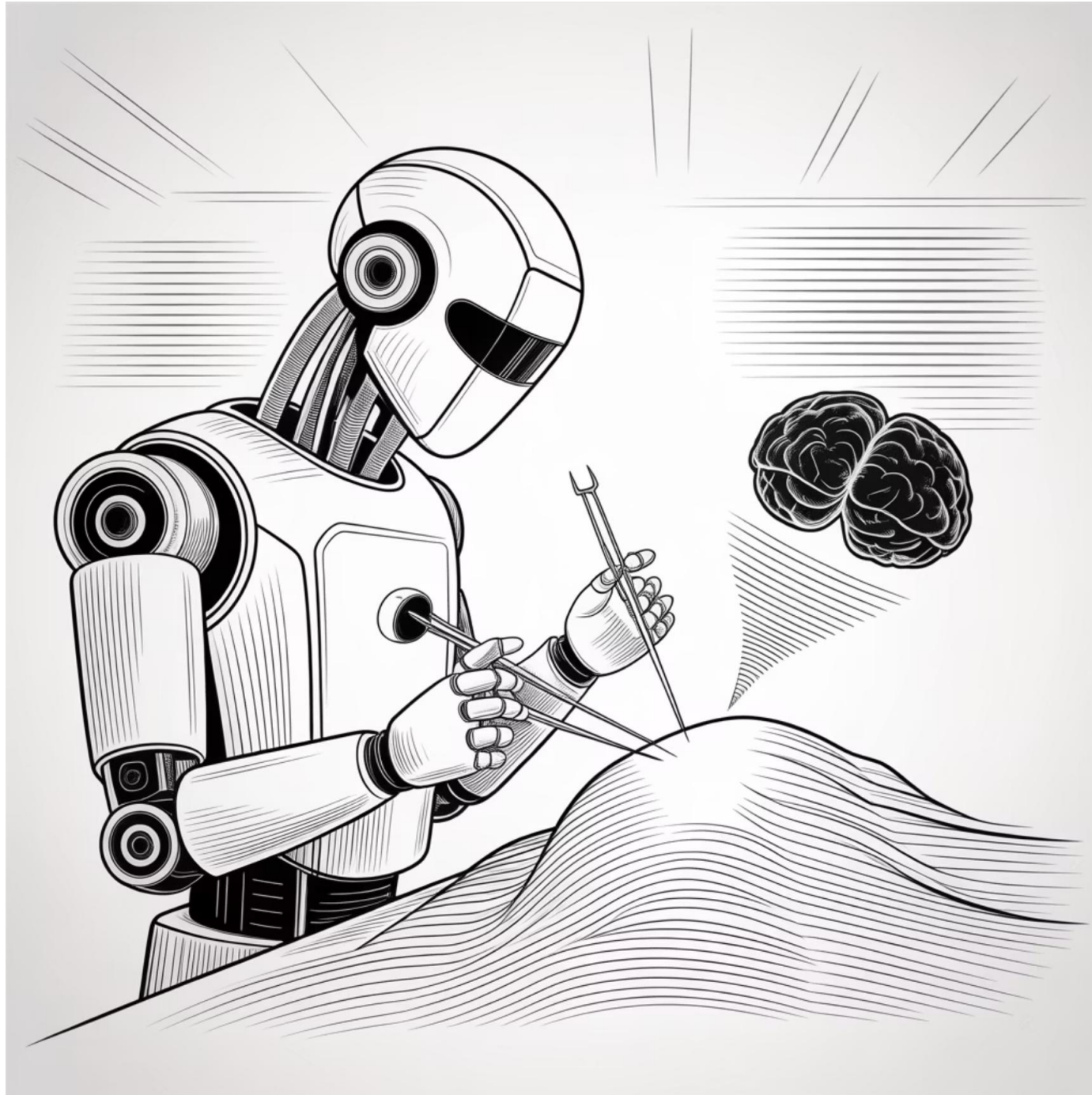
Algoritmos de aprendizado por reforço permitem que os veículos tomem decisões complexas em frações de segundo, como quando mudar de faixa com segurança ou como reagir a comportamentos inesperados de outros motoristas, sempre priorizando a segurança.



## Adaptação a Condições Adversas

Um dos maiores desafios da condução autônoma é a capacidade de adaptação a condições climáticas extremas. O aprendizado por reforço permite que os veículos ajustem continuamente seus modelos de comportamento conforme as condições mudam.

Empresas como Tesla, Waymo e Cruise estão na vanguarda da aplicação do aprendizado por reforço para criar veículos verdadeiramente autônomos capazes de navegar com segurança em situações para as quais nunca foram explicitamente programados.



## Transformando o Atendimento ao Paciente

O aprendizado por reforço está revolucionando a medicina em múltiplas frentes:

- **Dosagem Personalizada:** Sistemas que aprendem a ajustar tratamentos individualizados com base na resposta única de cada paciente, maximizando eficácia e minimizando efeitos colaterais
- **Robótica Cirúrgica:** Robôs que aprimoram sua precisão e técnica com cada operação realizada, aprendendo continuamente com a experiência acumulada
- **Descoberta de Medicamentos:** Aceleração dramática no desenvolvimento de novos fármacos através da exploração inteligente de espaços químicos virtualmente infinitos
- **Diagnóstico Adaptativo:** Sistemas que melhoram progressivamente sua capacidade de detectar doenças raras ou em estágios iniciais com base em feedback clínico

A precisão cirúrgica robótica melhora progressivamente graças ao aprendizado por reforço, permitindo procedimentos cada vez menos invasivos e com melhores resultados para os pacientes.

A aplicação do aprendizado por reforço na medicina não apenas melhora os resultados clínicos, mas também tem o potencial de democratizar o acesso a cuidados de saúde especializados em regiões carentes.

# Finanças Inteligentes: Otimização Contínua

## Trading Algorítmico

O aprendizado por reforço revolucionou o trading algorítmico ao criar sistemas que se adaptam dinamicamente às mudanças de mercado. Diferente dos algoritmos tradicionais com regras fixas, agentes de RL podem:

- Identificar padrões emergentes em tempo real
- Ajustar estratégias conforme a volatilidade do mercado
- Otimizar execução de ordens para minimizar impacto no mercado
- Equilibrar objetivos de curto e longo prazo

## Gestão de Portfólio

Na gestão de investimentos, algoritmos de RL superam abordagens estáticas ao:

- Ajustar alocação de ativos em resposta a mudanças macroeconômicas
- Balancear dinamicamente risco e retorno baseado em condições de mercado
- Incorporar múltiplas fontes de informação não-estruturada
- Aprender com decisões passadas para melhorar resultados futuros

## Detecção de Fraudes

Sistemas antifraude baseados em RL oferecem proteção superior por:

- Adaptarem-se continuamente a novas táticas fraudulentas
- Minimizarem falsos positivos que afetam clientes legítimos
- Identificarem padrões sutis de comportamento anômalo
- Equilibrarem segurança com experiência do usuário

# Indústria 4.0: A Revolução da Manufatura Inteligente

O aprendizado por reforço está transformando radicalmente o setor industrial, criando fábricas inteligentes que se auto-otimizam continuamente para máxima eficiência e produtividade.



## Otimização de Produção

Sistemas que aprendem a ajustar parâmetros de produção em tempo real, reduzindo desperdícios, consumo energético e tempo de inatividade. Os algoritmos de RL podem coordenar múltiplas máquinas simultaneamente para maximizar o throughput total da linha de produção.



## Manutenção Preditiva

Equipamentos que monitoram seu próprio desempenho e preveem falhas antes que ocorram, programando manutenção apenas quando necessário. Isto reduz drasticamente o tempo de inatividade não planejado e estende a vida útil dos equipamentos.



## Supply Chain Adaptativa

Redes de suprimentos que se reconfiguram dinamicamente em resposta a interrupções, mudanças de demanda ou gargalos logísticos. O RL permite que a cadeia de suprimentos aprenda a antecipar problemas e desenvolva estratégias alternativas automaticamente.

# Fundamentos Sólidos: K-Armed Bandits

Nossa jornada começa com um problema fundamental que captura a essência do aprendizado por reforço: o dilema entre **exploração** (buscar novas informações) e **aproveitamento** (utilizar o conhecimento atual).

O problema k-armed bandits pode ser visualizado como um cassino com k máquinas caça-níqueis, cada uma com uma probabilidade desconhecida de recompensa. O desafio é maximizar o ganho total decidindo quais máquinas jogar.

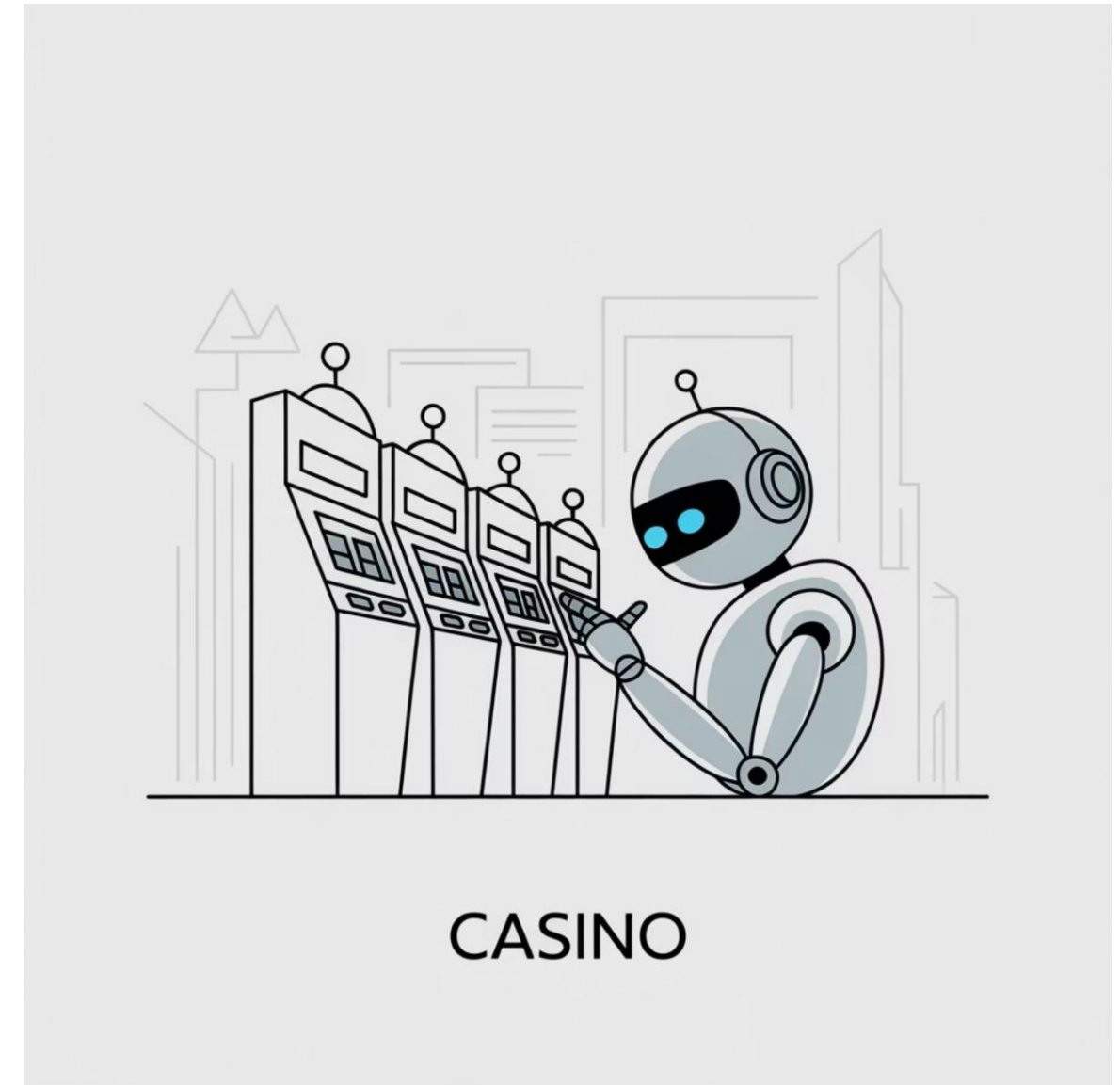
## Estratégias:

- **$\epsilon$ -greedy**: Escolhe a melhor máquina conhecida com probabilidade  $(1-\epsilon)$  e explora aleatoriamente com probabilidade  $\epsilon$
- **Softmax**: Seleciona máquinas com probabilidade proporcional à recompensa esperada
- **UCB (Upper Confidence Bound)**: Equilibra exploração e aproveitamento considerando a incerteza das estimativas

Este problema aparentemente simples está presente em inúmeras aplicações práticas:

Como um médico decide qual tratamento testar em um paciente

- Como plataformas de streaming escolhem quais filmes recomendar
- Como sites de e-commerce selecionam produtos para mostrar
- Como algoritmos de marketing decidem quais anúncios exibir





# Cadeias de Markov: A Memória Não Importa

Antes de abordarmos os MDPs completos, é fundamental compreender as Cadeias de Markov, que são sistemas onde o futuro depende apenas do estado presente, não do histórico de como chegamos até ele.

Matematicamente, uma Cadeia de Markov satisfaz a propriedade:  $P(X_{t+1} | X_t, X_{t-1}, \dots, X_0) = P(X_{t+1} | X_t)$

Esta propriedade, conhecida como "falta de memória" ou "propriedade Markoviana", é surpreendentemente útil para modelar diversos sistemas do mundo real:

- **Previsão do Tempo**

O clima de amanhã depende principalmente do clima de hoje, não de como estava há semanas

- **Sistemas de Filas**

O tempo de espera de um cliente depende apenas do número atual de pessoas na fila

- **Mercado Financeiro**

A hipótese de mercado eficiente sugere que os preços futuros dependem apenas dos preços atuais

- **Modelagem de Linguagem**

Modelos de Markov podem prever a próxima palavra baseando-se apenas nas palavras anteriores mais recentes

# O Mundo das Decisões Sequenciais

## Estado (s)

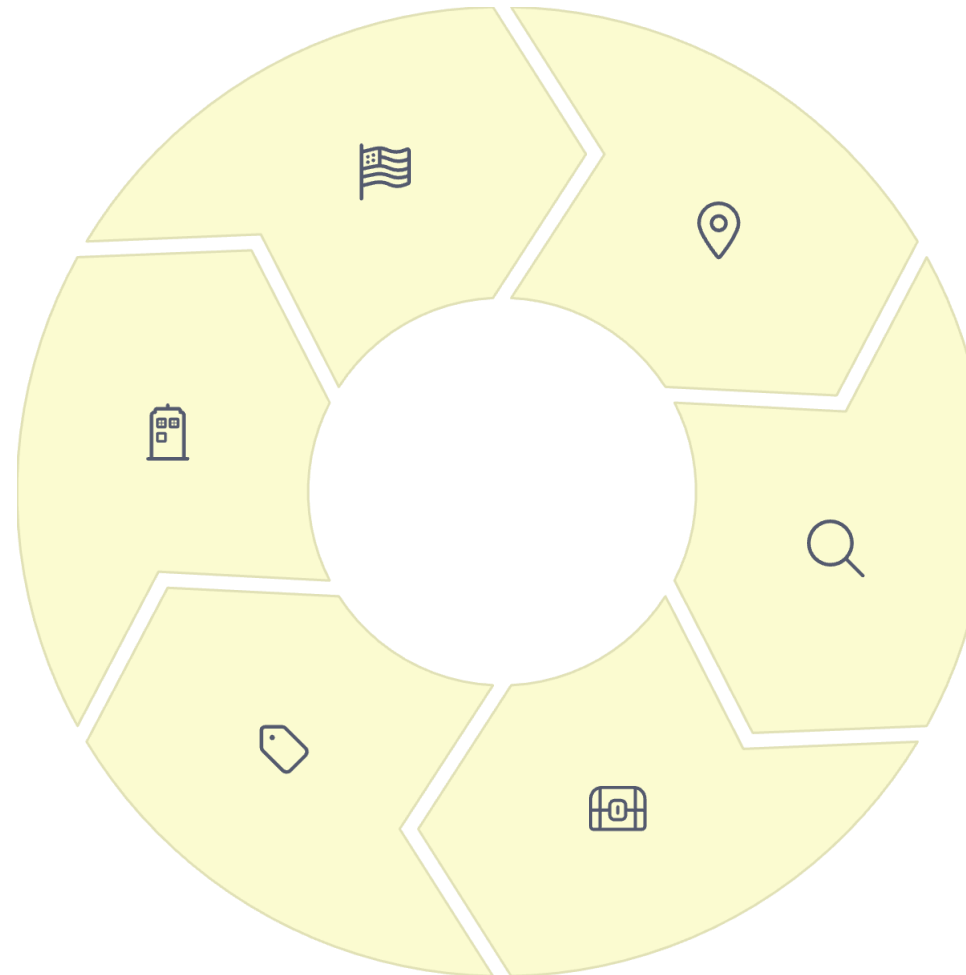
A representação completa da situação atual do ambiente. Em um jogo de xadrez, seria a posição de todas as peças no tabuleiro. Em um robô, poderia ser sua localização e os objetos ao seu redor.

## Política ( $\pi$ )

A estratégia  $\pi(a|s)$  que o agente usa para escolher ações em cada estado. O objetivo é encontrar a política ótima  $\pi^*$  que maximiza a recompensa total esperada.

## Desconto ( $\gamma$ )

Um fator entre 0 e 1 que determina a importância relativa de recompensas futuras versus imediatas. Valores próximos a 1 priorizam o longo prazo.



## Ação (a)

As escolhas disponíveis para o agente em cada estado. Podem ser discretas (mover para esquerda/direita) ou contínuas (aplicar uma força específica a um motor).

## Transição (P)

A probabilidade  $P(s'|s,a)$  de chegar ao estado  $s'$  após tomar a ação  $a$  no estado  $s$ . Captura a dinâmica do ambiente, que pode ser determinística ou estocástica.

## Recompensa (R)

O feedback imediato  $R(s,a,s')$  recebido após tomar a ação  $a$  no estado  $s$  e chegar ao estado  $s'$ . Guia o aprendizado do agente.

Os Processos de Decisão de Markov (MDPs) fornecem o framework matemático para formalizar problemas de tomada de decisão sequencial, onde cada ação influencia não apenas a recompensa imediata, mas todo o futuro do sistema.

# Processos de Decisão de Markov (MDPs)

Os MDPs estendem as Cadeias de Markov adicionando **ações** e **recompensas**, permitindo modelar problemas de decisão complexos. Um MDP é formalmente definido pela tupla  $(S, A, P, R, \gamma)$ :

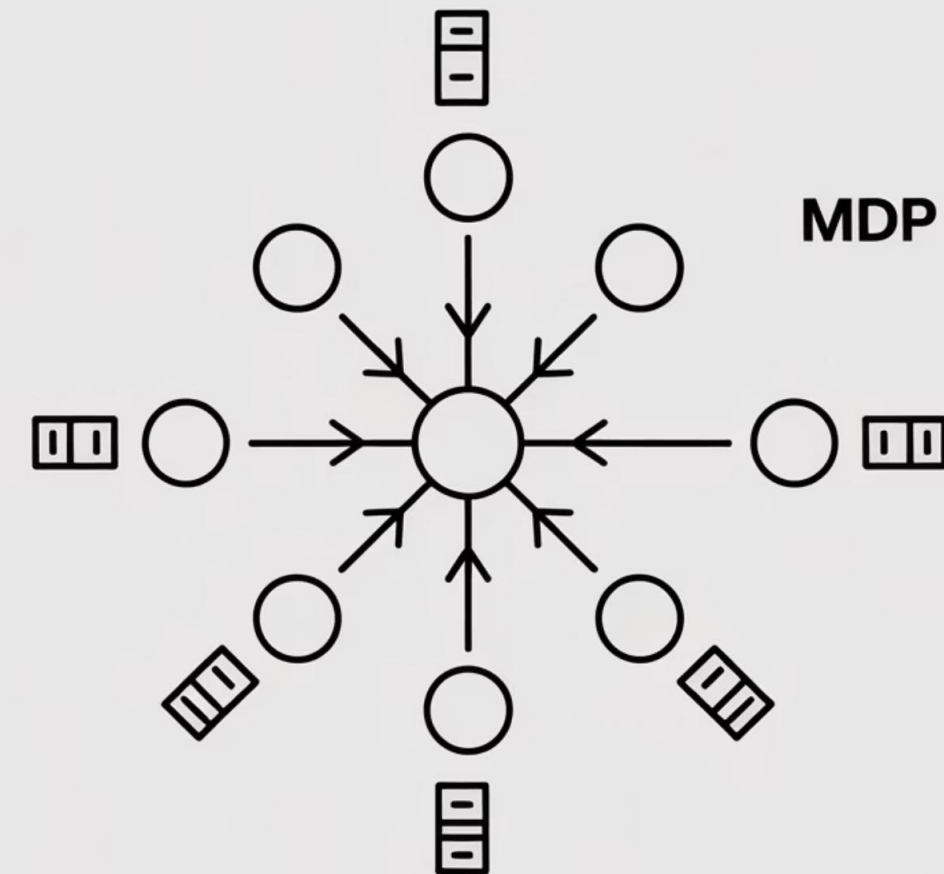
- **S**: Conjunto de estados possíveis
- **A**: Conjunto de ações disponíveis
- **P**: Função de probabilidade de transição  $P(s' | s, a)$
- **R**: Função de recompensa  $R(s, a, s')$
- **$\gamma$** : Fator de desconto para recompensas futuras

O objetivo em um MDP é encontrar uma política  $\pi^*$  que maximize a recompensa acumulada descontada esperada, dada por:

$$V^{\pi}(s) = E_{\pi} \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \mid s_0 = s \right]$$

Os MDPs capturam a essência de inúmeros problemas práticos:

- Como um robô deve navegar por um ambiente desconhecido
- Como um agente financeiro deve tomar decisões de investimento
- Como um sistema de recomendação deve selecionar itens para usuários
- Como um sistema de controle industrial deve ajustar parâmetros
- Como um assistente virtual deve responder a consultas de usuários



# Algoritmos Clássicos: Programação Dinâmica

A programação dinâmica oferece métodos para resolver MDPs quando conhecemos completamente o modelo do ambiente (P e R). Baseia-se nas poderosas Equações de Bellman, que estabelecem relações recursivas para valores ótimos.

Y

## Value Iteration

Inicializa  $V(s)$  arbitrariamente e itera até convergência:

$$V_{k+1}(s) = \max_a \sum_{s'} P(s' | s, a) [R(s, a, s') + \gamma V_k(s')]$$

A política ótima é extraída escolhendo ações que maximizam o valor esperado.

B

## Policy Iteration

Alterna entre avaliação de política (calcular  $V^\pi$ ) e melhoria de política:

1. **Avaliação:** Calcular  $V^\pi$  para a política atual  $\pi$
2. **Melhoria:** Atualizar  $\pi$  para ser gulosa em relação a  $V^\pi$

Converge para a política ótima em menos iterações que Value Iteration para muitos problemas práticos.



# Aprendizado na Incerteza: Monte Carlo

Quando não conhecemos o modelo do ambiente, precisamos aprender através da experiência. Os métodos de Monte Carlo estimam valores a partir de episódios completos de experiência, sem exigir conhecimento prévio das probabilidades de transição ou funções de recompensa.

## Como Funciona:

1. Gerar episódios completos seguindo a política atual
2. Para cada estado visitado, calcular o retorno (soma das recompensas de
3. Atualizar a estimativa de valor como a média dos retornos observados

A fórmula de atualização para o valor de um estado  $s$  é:

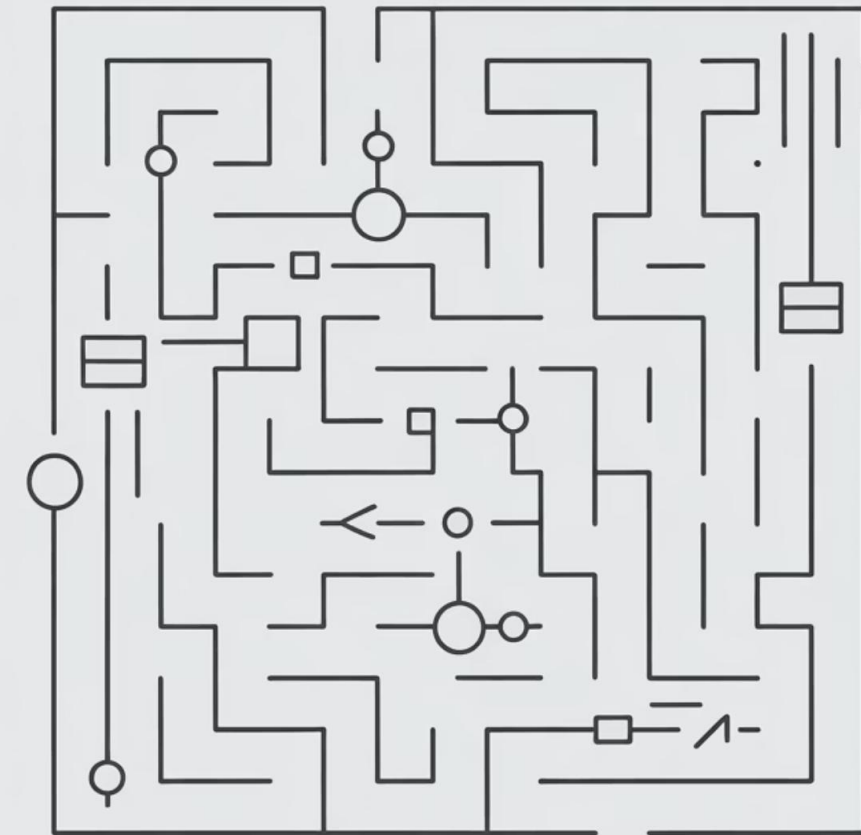
$$V(s) \leftarrow V(s) + \alpha [G_t - V(s)]$$

onde  $G_t$  é o retorno observado a partir do tempo  $t$  e  $\alpha$  é a taxa de aprendi

Os métodos de Monte Carlo são particularmente úteis em:

- Jogos com regras determinísticas mas estratégia complexa
- Simulações onde podemos gerar episódios completos
- Problemas onde é difícil modelar transições mas fácil simular
- Situações onde queremos aprender offline a partir de dados históricos

A principal limitação é a necessidade de episódios completos, o que pode ser impraticável em tarefas contínuas ou muito longas.





# Q-Learning: O Santo Graal do Aprendizado Off-Policy

O Q-Learning é um dos algoritmos mais importantes do aprendizado por reforço, permitindo aprender a política ótima independentemente da política de exploração utilizada durante o treinamento.

## A Equação de Atualização

$$Q(s, a) \leftarrow Q(s, a) + \alpha [r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$$

Esta equação combina a recompensa imediata  $r$  com o valor máximo possível do próximo estado  $s'$ , descontado por  $\gamma$ . A diferença entre esta soma e o valor atual  $Q(s, a)$  é o erro de temporal difference, usado para atualizar a estimativa.

## Características Principais

- **Off-policy:** Aprende a política ótima mesmo seguindo uma política exploratória
- **Bootstrapping:** Atualiza estimativas baseando-se em outras estimativas, sem esperar o fim do episódio
- **Convergência:** Garante convergência para a política ótima sob certas condições
- **Exploration-exploitation:** Tipicamente implementado com estratégias como  $\epsilon$ -greedy

O Q-Learning é como ter um assistente que aprende a estratégia perfeita observando suas decisões imperfeitas - uma propriedade extremamente poderosa que permite exploração segura em ambientes complexos.

# SARSA e Expected SARSA: A Família On-Policy

## SARSA (State-Action-Reward-State-Action)

Diferente do Q-Learning, o SARSA é um algoritmo on-policy que aprende sobre a política que está atualmente seguindo:

$$Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma Q(s', a') - Q(s, a)]$$

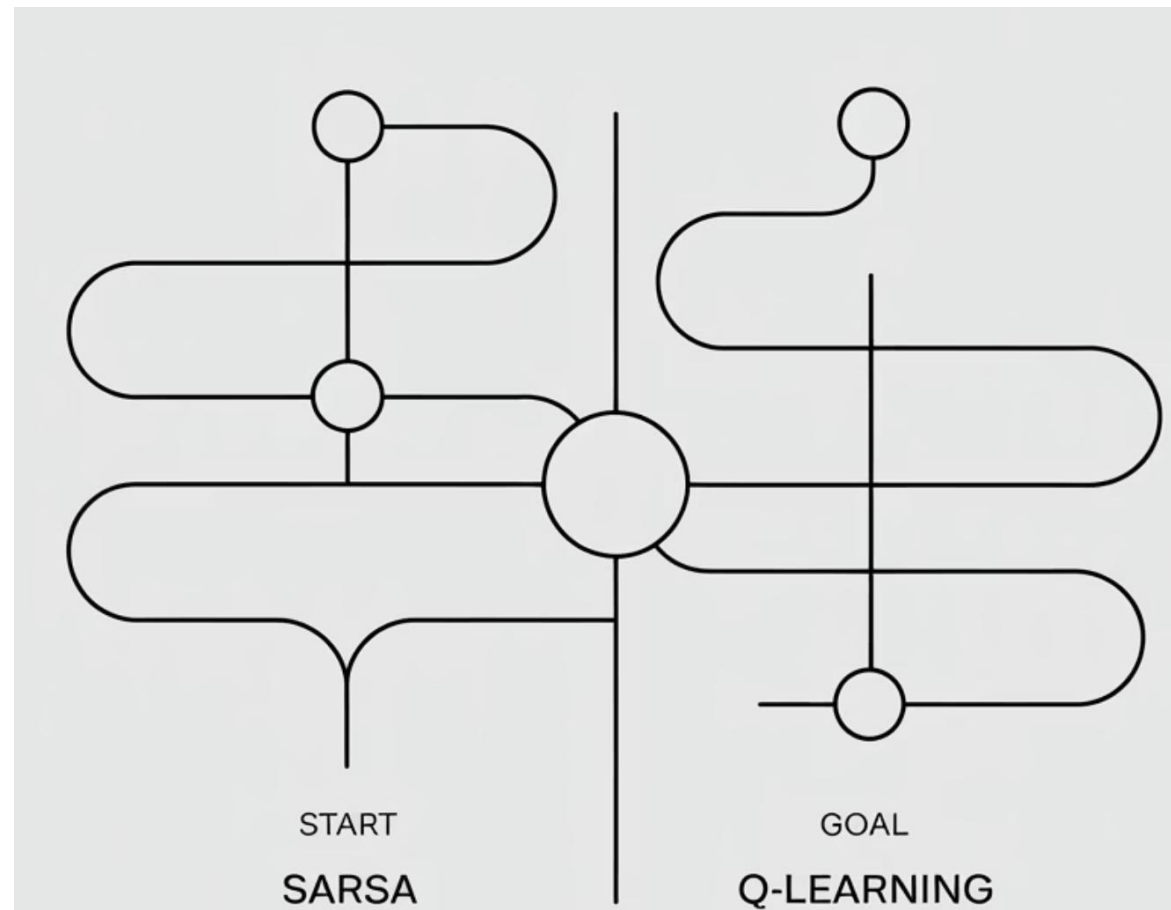
A diferença crucial é que  $a'$  é a ação realmente escolhida no próximo estado, não necessariamente a melhor ação possível. Isto torna o SARSA mais conservador e cauteloso em ambientes com riscos, pois considera as imperfeições da política de exploração atual.

## Expected SARSA

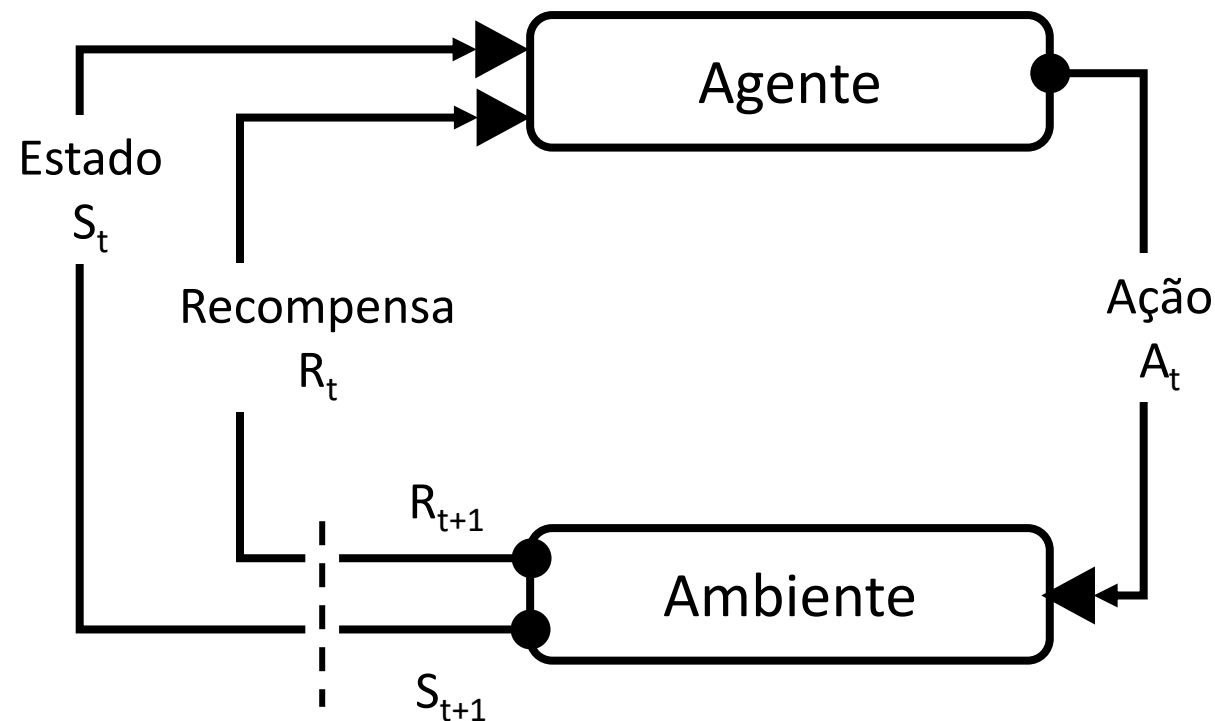
Uma elegante ponte entre Q-Learning e SARSA, o Expected SARSA usa a expectativa sobre todas as ações possíveis:

$$Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \sum_{a'} \pi(a' | s') Q(s', a') - Q(s, a)]$$

Esta abordagem reduz a variância das atualizações e frequentemente apresenta melhor desempenho que ambos Q-Learning e SARSA, combinando a estabilidade do SARSA com a eficiência do Q-Learning.



# A estrutura básica do Processo de Decisão de Markov (MDP)



- O ponto de partida do RL é a **declaração do problema**,
- Começa com o **agente** (o que aprende e toma ações) e o **ambiente**
- No momento  **$t$**  o ambiente está num estado ( **$S_t$** ).
- o agente toma uma **ação ( $A_t$ )**, que é recebida pelo ambiente.
- O ambiente, por sua vez, produz uma **recompensa ( $R_{t+1}$ )** e um novo **estado ( $S_{t+1}$ )** no próximo momento ( **$t+1$** ).
- Esse processo se repete.

# Bibliografia

## Bibliografia Básica

**FACELI, Katti et al.** Inteligência Artificial: Uma Abordagem de Aprendizado de Máquina. Rio de Janeiro: LTC, 2021.

**NETTO, Amilcar; MACIEL, Francisco.** Python para Data Science e Machine Learning Descomplicado. Rio de Janeiro: Alta Books, 2021.

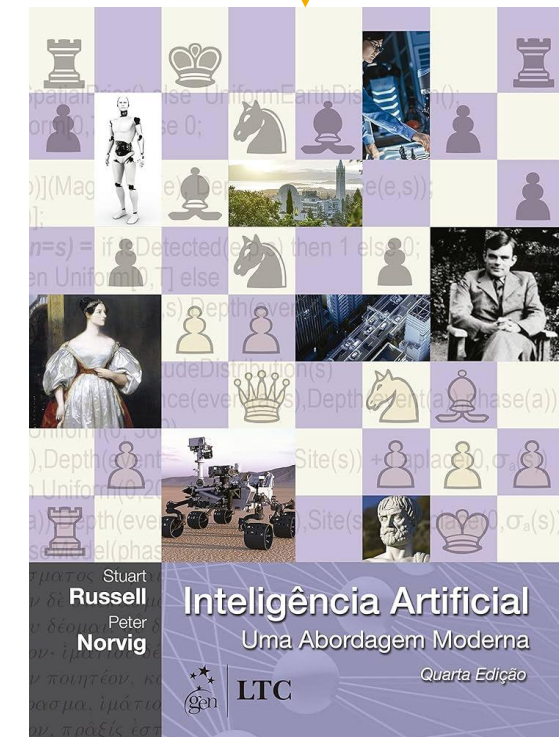
**SÁ, Y. V. A.** Desenvolvimento de aplicações IA: Robótica, Imagem e Visão Computacional. São Paulo: Platos, 2021.

## Bibliografia Complementar

**SUTTON, Richard; BARTO, Andrew.** Reinforcement Learning: An Introduction. Cambridge: MIT Press, 2018.

**RUSSELL, Stuart J.; NORVIG, Pete.** Inteligência Artificial: Uma Abordagem Moderna. Rio de Janeiro: GEN LTC, 2022.

**SILVA, Fabrício M. et al.** Inteligência artificial. Porto Alegre: SAGAH, 2018.



## Reinforcement Learning

An Introduction  
second edition

Richard S. Sutton and Andrew G. Barto

Disponível para download (pelo autor)) em:  
<http://incompleteideas.net/book/the-book.html>

Além dos livros listados, compartilharemos ao longo do curso artigos científicos recentes, tutoriais online e recursos complementares para aprofundamento em tópicos específicos.

# Pré Work próxima Aula

## Baixe este livro

SUTTON, Richard; BARTO, Andrew.  
Reinforcement Learning: An  
Introduction. Cambridge: MIT Press,  
2018.

Disponível para download (pelo autor)) em:  
<http://incompleteideas.net/book/the-book.html>

Leia o Capítulo 2 .



# Pré Work próxima Aula

## Baixe este livro

SUTTON, Richard; BARTO, Andrew.  
Reinforcement Learning: An  
Introduction. Cambridge: MIT Press,  
2018.

Disponível para download (pelo autor) em:  
<http://incompleteideas.net/book/the-book.html>

Leia o Capítulo 2 .

# Acesso materiais de Aula

[https://github.com/DeepFluxion/IBMEC\\_Aprendizado\\_Reforco\\_2026](https://github.com/DeepFluxion/IBMEC_Aprendizado_Reforco_2026)

The screenshot shows a web browser window displaying the GitHub repository page for 'IBMEC\_Aprendizado\_Reforco' by user 'DeepFluxion'. The browser's address bar shows the URL 'github.com/DeepFluxion/IBMEC\_Aprendizado\_Reforco'. The repository page includes a header with the repository name, a search bar, and navigation links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. Below the header, the repository is listed as 'Public' with 0 watches, 0 forks, and 0 stars. The main content area shows a file list with columns for file name, description, and time ago. The files listed are 'Aula\_0' (Criado usando o Colab, 6 minutes ago), 'Aula\_1' (Create gitnore, 12 minutes ago), and 'README.md' (Initial commit, 19 minutes ago). The 'README' file is selected, showing the title 'IBMEC\_Aprendizado\_Reforco'. On the right side, there are sections for 'About' (No description, website, or topics provided), 'Releases' (No releases published, Create a new release), 'Packages' (No packages published, Publish your first package), and 'Languages'.

Prima Permitir x Tradução do Capítulo 2 de RLbo x git - Como criar uma pasta den x Aula\_0\_Revisão de Probailidade x DeepFluxion/IBMEC\_Aprendiza x +

github.com/DeepFluxion/IBMEC\_Aprendizado\_Reforco

Sites Sugeridos Importado do IE Google Inteligência Artificial... Widgets Avaliação da relaça... Rafael Santos Como instalar uma... Resultado de image... Todos os favoritos

DeepFluxion / IBMEC\_Aprendizado\_Reforco

Type to search

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

IBMEC\_Aprendizado\_Reforco Public

Pin Watch 0 Fork 0 Star 0

main 1 Branch 0 Tags

Go to file Add file <> Code

DeepFluxion Criado usando o Colab 53c44de · 6 minutes ago 11 Commits

Aula_0	Criado usando o Colab	6 minutes ago
Aula_1	Create gitnore	12 minutes ago
README.md	Initial commit	19 minutes ago

README

## IBMEC\_Aprendizado\_Reforco

About

No description, website, or topics provided.

- Readme
- Activity
- 0 stars
- 0 watching
- 0 forks

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

Languages

https://github.com/DeepFluxion/IBMEC\_Aprendizado\_Reforco

Pesquisar

POR PTB2 20:00 07/08/2025