Introdução à Inteligência Artificial
Agentes Inteligentes

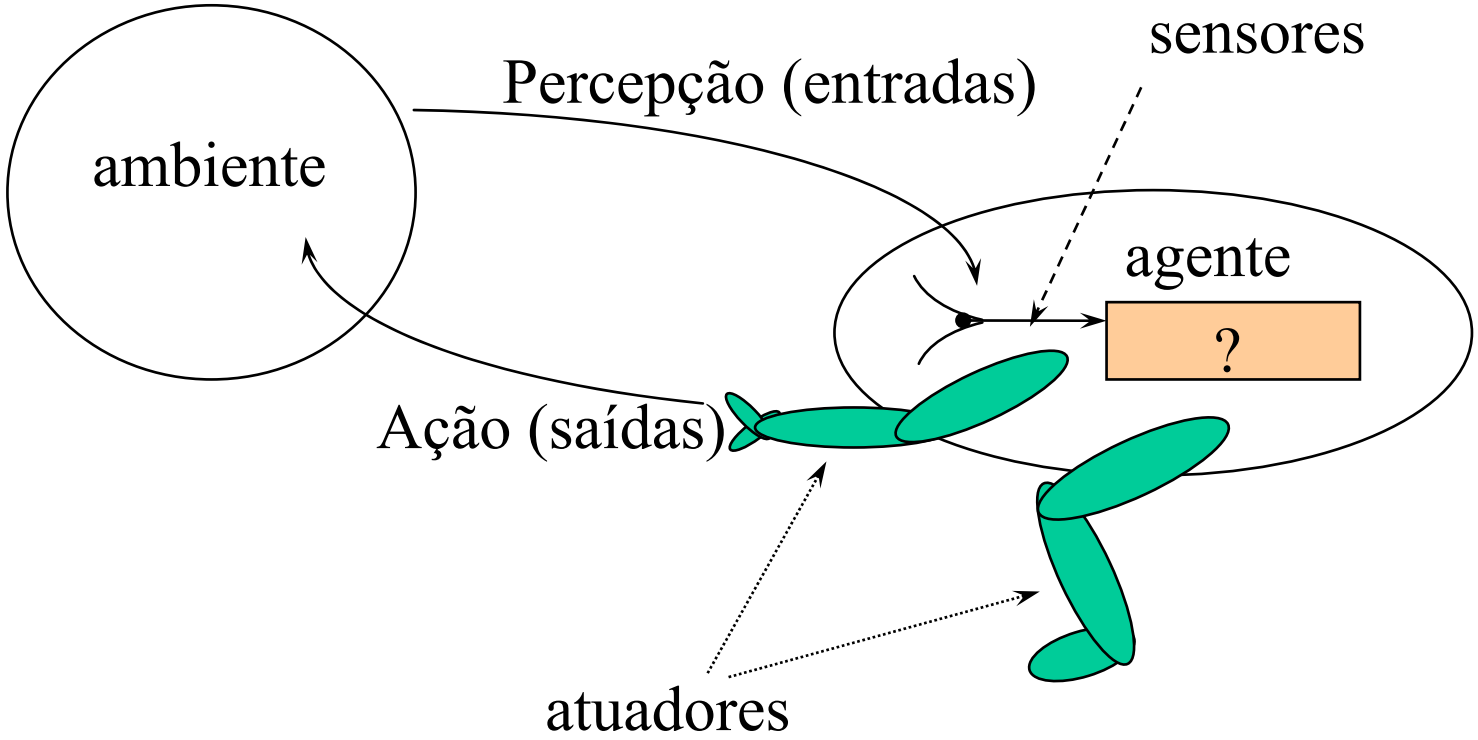
Leliane Nunes de Barros

leliane@ime.usp.br

O que é Inteligência Artificial?

- O estudo e a construção de “sistemas computacionais inteligentes”
- Nossa definição de “sistemas inteligentes”: sistemas que pensam e agem racionalmente
- Em IA sistemas inteligentes são vistos (modelados) como ***agentes inteligentes***

Agente em IA



Objetivo de IA: escrever o *programa do agente*

Agente

- **Percebe** seu ambiente através de **sensores** e age sobre o ambiente através de atuadores
- Agente humano:
  - **Sensores**: olhos, ouvidos, …
  - **Atuadores**: mãos, pernas, boca, …
- Agente robótico:
  - **Sensores**: câmeras, detector infra-vermelho
  - **Atuadores**: vários tipos de motores, e.g. garra de um robô.
- E um agente de software?

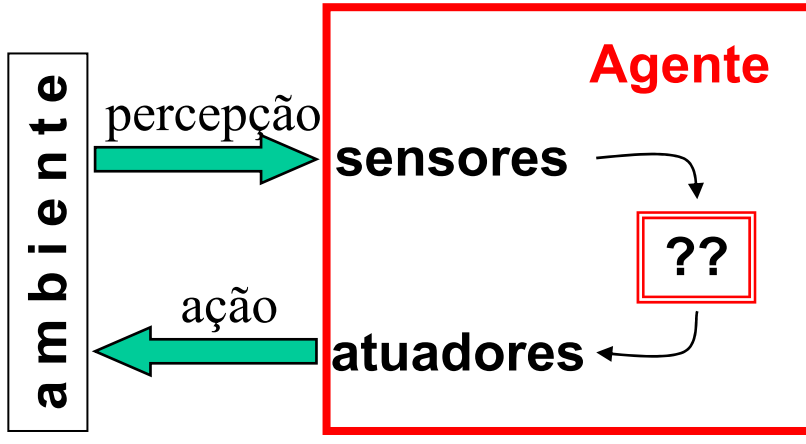
Agente em IA

Agente é qualquer entidade que:

- **percebe** seu ambiente através de sensores (ex: arquivos de imagem, vídeo e audio, entrada pelo teclado, conteúdo de arquivos ou de BD, páginas Web, etc ...)
- **age** sobre o ambiente através de atuadores (ex: envio de instruções para um atuador ou escrita em arquivos)

Relação entre ambiente e agente

- ambiente físico/robôs
- ambiente de software/softbots



Agente onisciente

- agente que percebe quando executa ações
- agente que conhece todos os efeitos de suas ações
  - em certos ambientes é impossível modelar completamente a realidade
  - relaxar os critérios de desempenho de um agente ==> ambientes artificiais

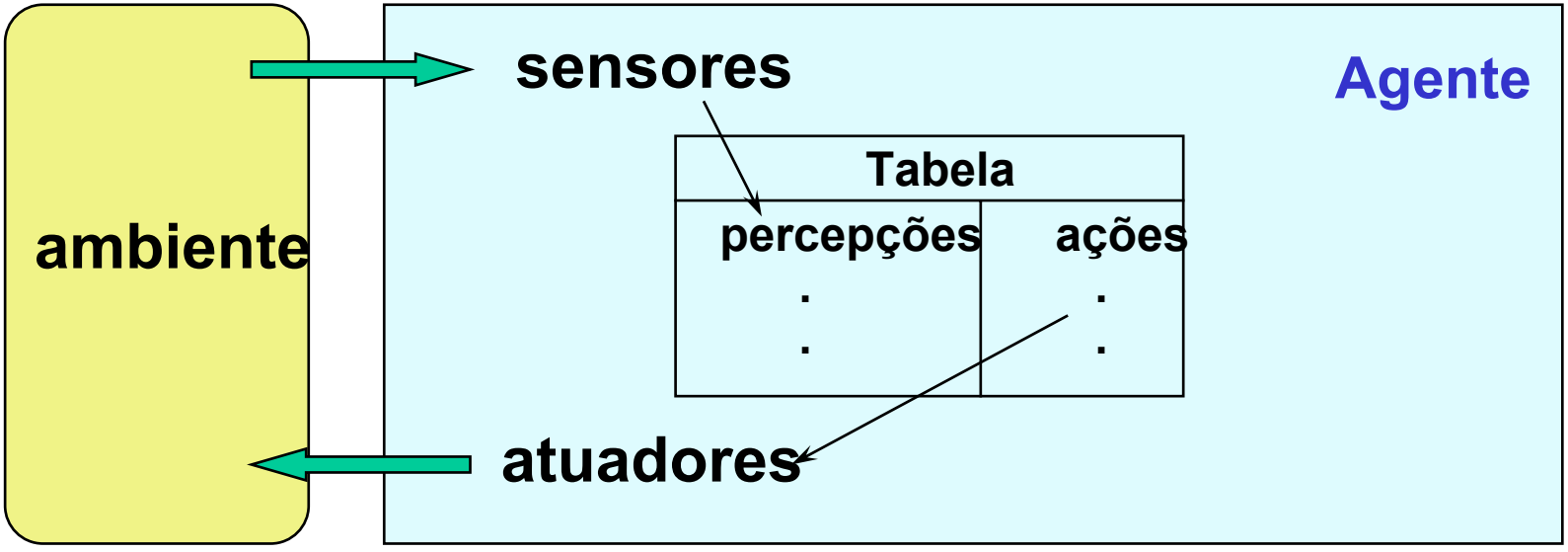
Agente autônomo

- capacidade de interagir com o ambiente e extrair informações sobre o mundo
  - um agente autônomo possui algum conhecimento inicial e a habilidade de inferir ou aprender novos conhecimentos
- o comportamento do agente pode depender de dois fatores: do conhecimento embutido em seu programa e de sua própria experiência

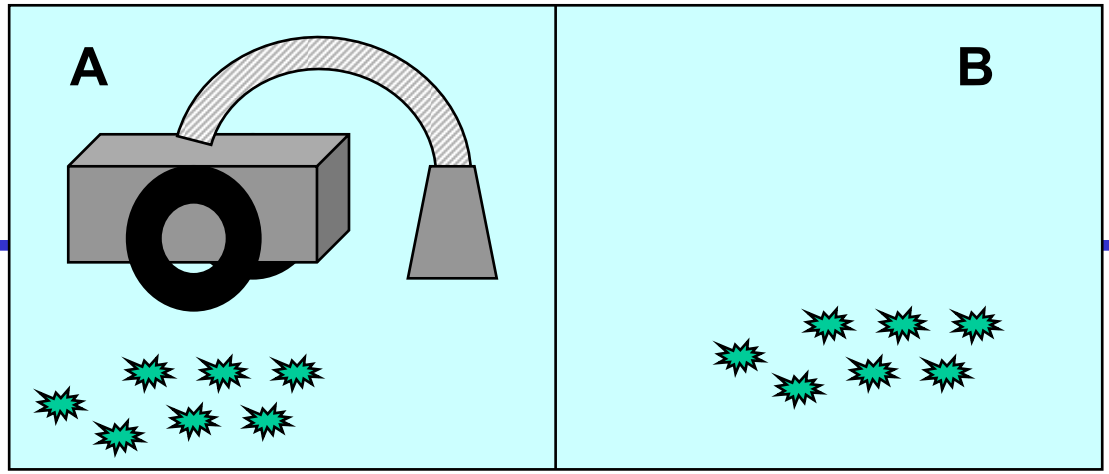
Mapeamento ideal entre seqüências de percepções e ações

- mapeamento percepção/ação: tabela das ações que o agente toma em resposta a cada possível seqüência de percepções
- cada mapeamento percepção/ação descreve um tipo diferente de agente
- mapeamentos ideais descrevem agentes ideais
- é possível especificar um mapeamento sem a enumeração exaustiva

Agente que consulta uma tabela



- Limitações
  - Mesmo para problemas simples ➔ tabelas muito grandes
    - ex. xadrez 30^100
  - Nem sempre é possível, ou por ignorância ou por limitação de tempo e espaço, construir a tabela



PERCEPÇÃO [sala, estado]	AÇÃO
[A, limpo]	Ir para a direita
[A, sujo]	Aspirar
[B, limpo]	Ir para a esquerda
[B, sujo]	Aspirar
[A, limpo], [A, limpo]	Ir para direita
[A, limpo], [A, sujo]	Aspirar
...	...
[A, limpo], [A, limpo], [A,limpo]	Ir para a direita

Medida de Desempenho

- Critério que define o grau de desempenho de um agente na realização de uma dada tarefa
  - a escolha errada da medida de desempenho pode acarretar num comportamento indesejado

Qual é a melhor medida de desempenho para o aspirador de pó?

- Maximizar a quantidade de sujeira aspirada em 8 horas ou
- Minimizar a quantidade de sujeira no chão em 8 horas
- Maximizar a área coberta pelo aspirador

Agente racional

- Agente que para cada seqüência de percepções possível, seleciona uma ação que ele espera que maximize sua medida de desempenho.
- Agente que "faz a coisa certa" enquanto age em seu ambiente
  - a ação correta é aquela que faz o agente ter o melhor desempenho
- **Questão**: *como* e *quando* avaliar o desempenho do agente?
    - *como*: através da definição de uma medida de desempenho objetiva
    - *quando*: ao longo das tarefas realizadas pelo agente

Racionalidade depende ...

- (1) da medida de desempenho que define o sucesso do agente
- (2) da seqüência de percepções do agente
- (3) do que o agente sabe sobre o ambiente
- (4) das ações que o agente pode realizar

Agente racional ideal: para cada seqüência de percepções o agente escolhe a ação que maximiza seu desempenho baseado nas informações de percepção e de seu conhecimento sobre o mundo

Estrutura de agentes inteligentes

- IA se preocupa em projetar o **programa do agente:** função que implementa o mapeamento entre percepção e ação
- O programa do agente roda em uma arquitetura: dispositivo de computação que inclue sensores e atuadores.

$$agente = arquitetura + programa$$

- Componentes de especificação de agentes: PEAS (*Performance, Environment, Actuators, Sensors*)

Especificação da tarefa do agente:

- ***P - P****erformance*
- ***E - E****nvironment*
- ***A - A****ctuators*
- ***S - S****ensors*

Exemplos de ambientes de tarefa

Agente	*percepções*	*ações*	*Medida de desempenho*	*Ambiente*
Diagnóstico médico	Sintomas, resultados de exames, ...	Perguntar, realizar ou prescrever exames, ...	Maximizar a saúde do paciente, minimizar custos	Paciente, consultório, Laboratório, ...
Análise de imagens de satélite	Pixels	imprimir uma classificação	classificar corretamente	Imagens de satélite
Tutor de português	Palavras digitadas	Imprimir exercícios, sugestões, correções, ...	Melhorar o desempenho do estudante	Conjunto de estudantes
Filtro de *emails*	mensagens	Aceitar ou rejeitar mensagens	Aliviar a carga de leitura do usuário	Mensagens, usuários
Motorista de taxi	Imagens, velocímetro, sons	brecar, acelerar, virar, falar com passageiro, ...	Segurança, rapidez, economia, conforto,...	Ruas, pedestres, carros, ...
Músico de jazz	Sons, seus e de outros músicos, grades de acordes	Escolher e tocar notas no andamento	Tocar bem, se divertir, agradar	Músicos, público, grades de acordes

Programa do agente

Agente baseado em tabela

função Agente-olha-tabela (*percepção*) **devolve** *ação*

estático: *percepções,* uma seqüência inicialmente vazia

tabela, uma tabela indexada por todas as seqüências possíveis de percepções, inicialmente completamente especificada

append *percepção* no final da seqüência de *percepções*

ação ← LOOCKUP*(percepções, tabela)*

devolve *ação*

Agente baseado em tabela: dificuldades

- Número muito grande de entradas na tabela: *levaria muito tempo para o projetista construir a tabela*
- Agente sem autonomia: decisões compiladas na fase de projeto; o agente se perde diante de qualquer mudança no ambiente

Porque um agente que "raciocina", em oposição a olhar uma tabela do tipo percepção/ação, pode evitar as dificuldades acima?

Exemplo:
agente motorista de taxi

Tipo de agente	Percepção	Ações	Goals	Ambiente
Motorista de Taxi	Câmeras controláveis, velocímetro, odômetro, sonar, microfone, GPS	Virar as rodas, acelerar, freiar, conversar com o passageiro	Viagem segura, rápida e confortável. Maximizar lucros.	Estradas, tráfego de outros veículos, pedestres, fregueses.

Medidas de desempenho:

- **chegar ao local correto**
- **minimizar o gasto de combustível**
- **minimizar o custo da viajem**
- **minimizar violações de trânsito**
- **maximizar a segurança e conforto do passageiro**
- **maximizar os lucros**

Cinco tipos de programas de agentes

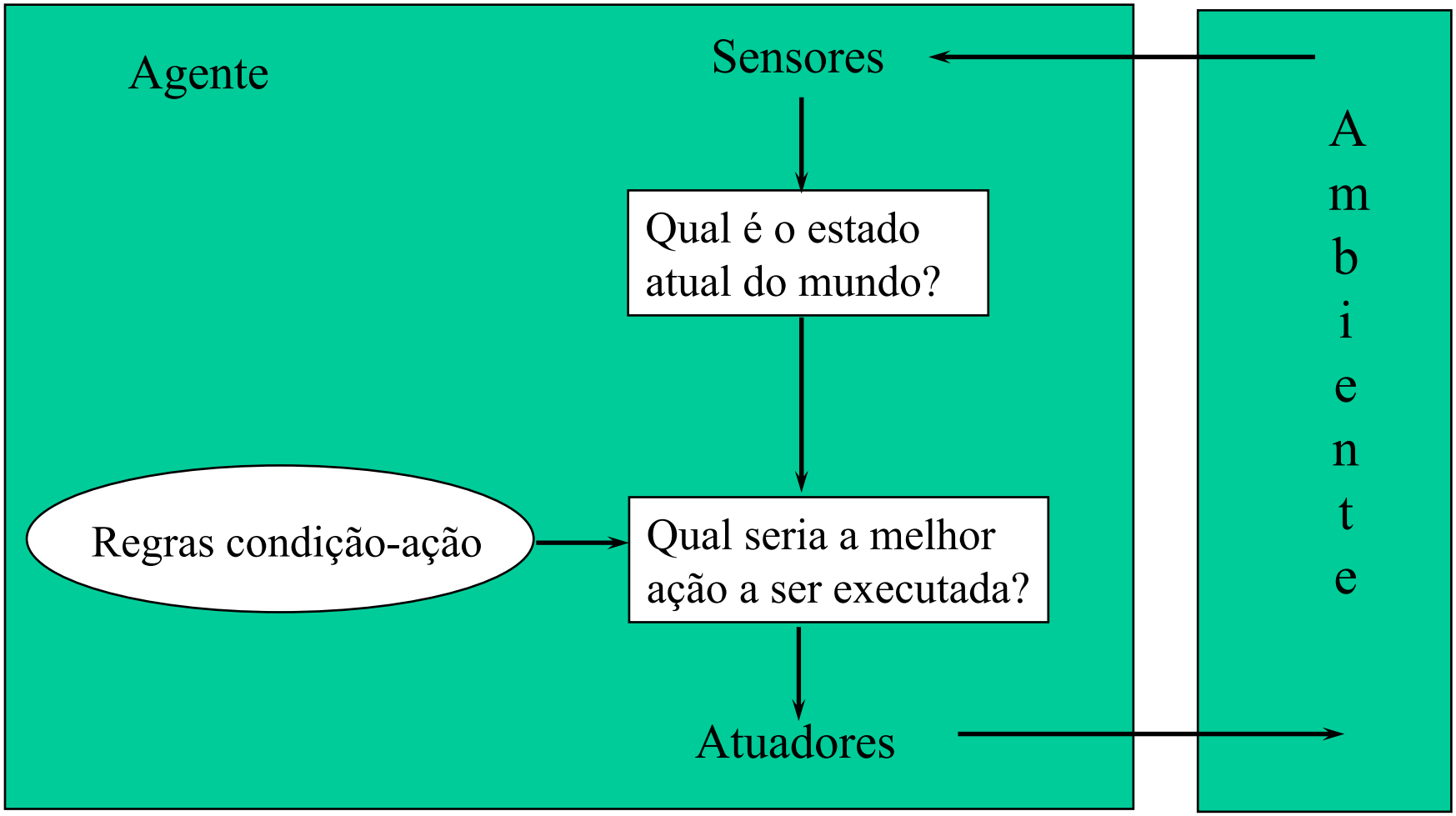
- Agente Reativo Simples
- Agente Reativo baseado em Modelo
- Agente Baseado em Meta
- Agente Baseado em Utilidade
- Agente Aprendiz

autonomia

Agente reativo

- Motorista de taxi:
  - Percepção visual: 50 MB/sec ==> **Tabela de Consulta** para uma hora de carro em movimento: $\sim$ $\mathbf{2^{60 \times 60 \times 50M}}$ entradas
- Porções da tabela podem ser sumarizadas
- Regra **CONDIÇÃO-AÇÃO**
  - Também chamada de **regra situação-ação** ou **regras de produção** ou **regras se-então**
  - Ex.: **Se** carro-em-frente-freia **Então** inicia-freiar

Agente reativo baseado em regras



Se carro-da-frente-freia ***Então*** comece-a-frejar

Agente reativo baseado em regras

função Agente-Reativo-Simples (*percepção*) **devolve** *ação*

estática: *regras,* um conjunto de regras condição-ação

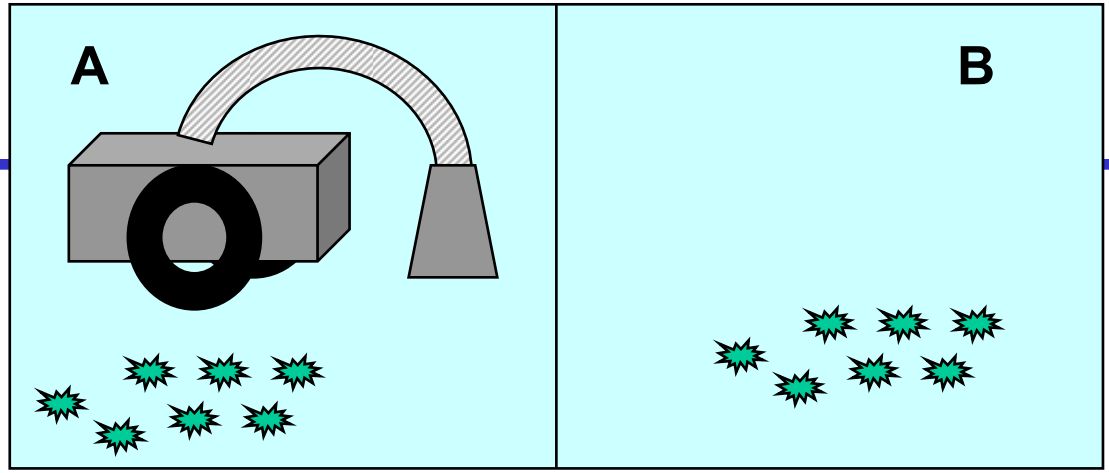
estado ← INTERPRETA-ENTRADA(*percepção*)

regra ← CASA-REGRA *(estado, regras)*

ação ← AÇÃO-DA-REGRA*(regra)*

devolve *ação*

Descrição abstrata do estado do mundo a partir da percepção



function Agente-Reativo-Aspirador ([local, status]) **return** uma ação

if status=Sujo **then return** Aspira
else if local=A **then return** Direita
else if local=B **then return** Esquerda

Agente reativo baseado em regras

- Vantagens e desvantagens
  - Regras condição-ação: representação inteligível, modular e eficiente
  - Mas ainda pode sofrer das mesmas limitações da tabela

  (em alguns casos, uma representação concisa pode torná-lo mais eficiente) ➔ lições da área de Representação de Conhecimento

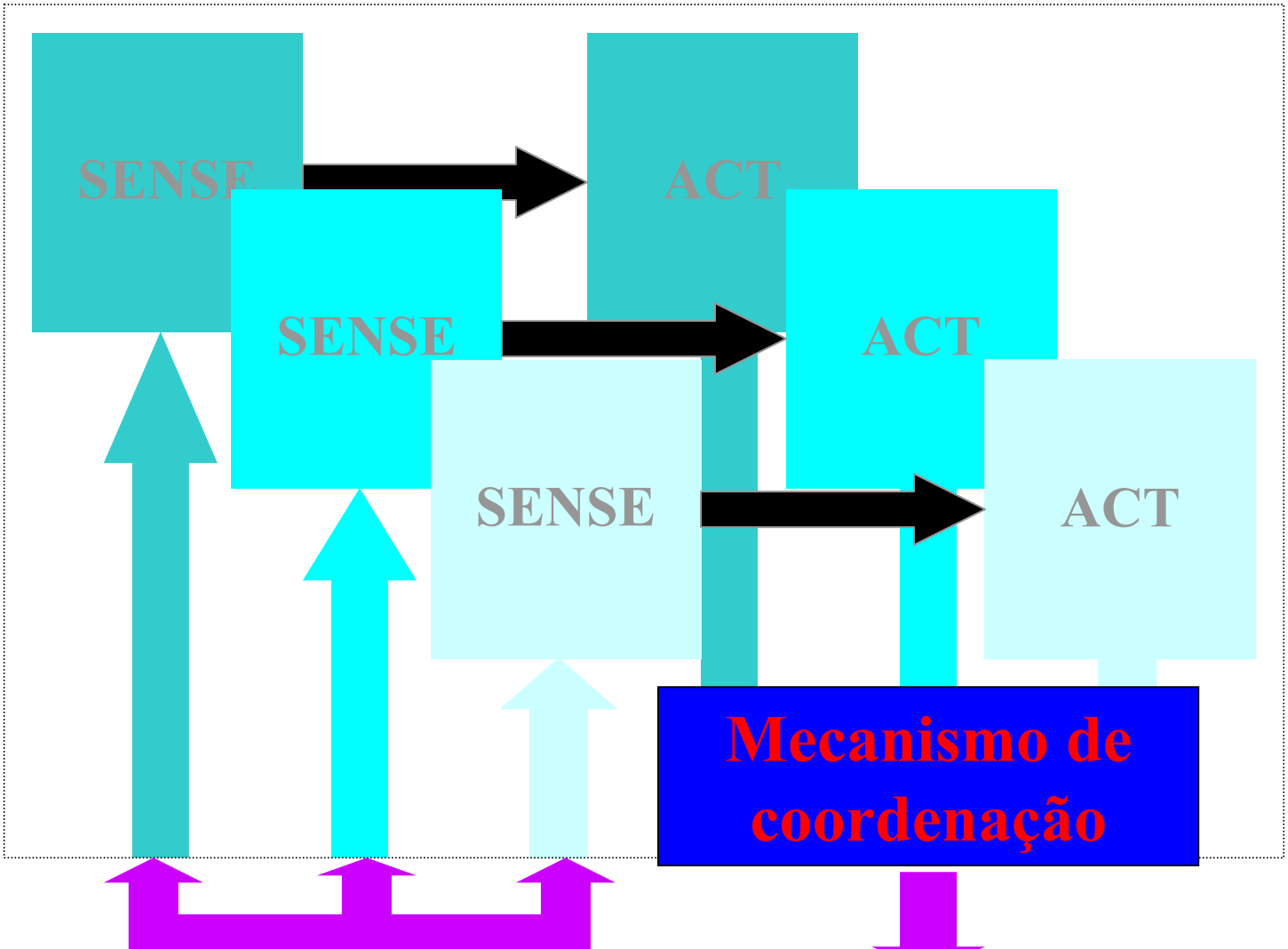
Arquiteturas reativas para robôs

- Combinam vários tipos diferentes de agentes reativos simples (também chamados de diferentes *comportamentos reativos*)
- Podem realizar tarefas complexas através do comportamento global dos agentes reativos

Arquiteturas reativas para robôs

- Surgidas no final dos anos 80.
- Fundamentadas em estudos do comportamento animal (Etologia) ➔ baseada em comportamentos.
- Baseadas em processamento paralelo (vários comportamentos simultaneamente ativos).

Arquitetura Reativa



Mecanismos de coordenação

- **Coordenação Competitiva:** a ação resultante num dado instante é selecionada a partir de uma competição entre os comportamentos ativos (um vence).
- **Coordenação Cooperativa**: a função de coordenação produz uma ação resultante para a qual contribuem todos os comportamentos ativos.

Arquitetura REACT

Comportamentos Reativos para Robôs Móveis (LTI – Poli)

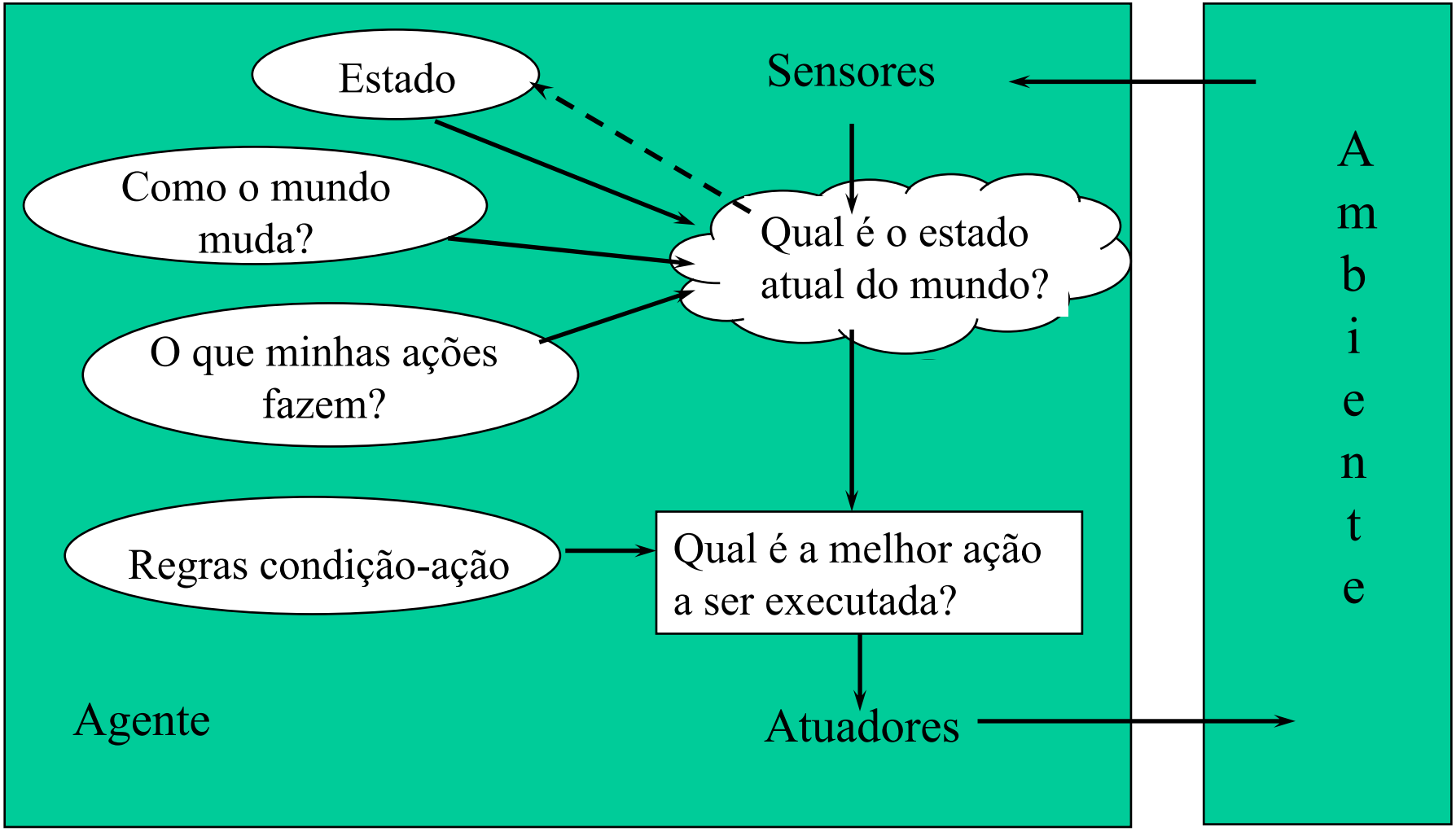
Agentes reativos baseados em modelo

- agentes que possuem percepção parcial do ambiente podem acompanhar suas mudanças através de uma representação interna do estado do mundo
- percepções isoladas não fornecem acesso ao estado completo do mundo
  - existem estados do mundo diferentes que fornecem a mesma percepção
  - o agente necessita manter informação interna para distinguir estados do mundo aparentemente iguais

Necessidade de um modelo

- Para "imaginar a parte do mundo que não está observável no momento" o agente mantém um ***estado interno*** que depende do histórico de percepções.
- Neste novo agente, para determinar como o mundo está num determinado momento, ele usa:
  - informações perceptuais atuais (como o agente reativo)
  - seu estado interno
  - informações a respeito de como o mundo evolui independentemente de suas ações (***modelo do mundo***)
  - informações a respeito do impacto/efeito de suas próprias ações no mundo e, com isso, atualiza seu estado interno**.**

Agente Reativo baseados em modelo



Agente Reativo com Estado Interno

unção Agente-Reativo-com-Memória (*percepção*) **devolve** *ação*

estática: *estado*, descrição interna do estado do mundo

regras, conjunto de regras **condição-ação**

ação, a última ação executada, inicialmente nula

estado ← ATUALIZA-ESTADO (*estado, ação, percepção*)

regra ← CASA-REGRA (*estado, regras)*

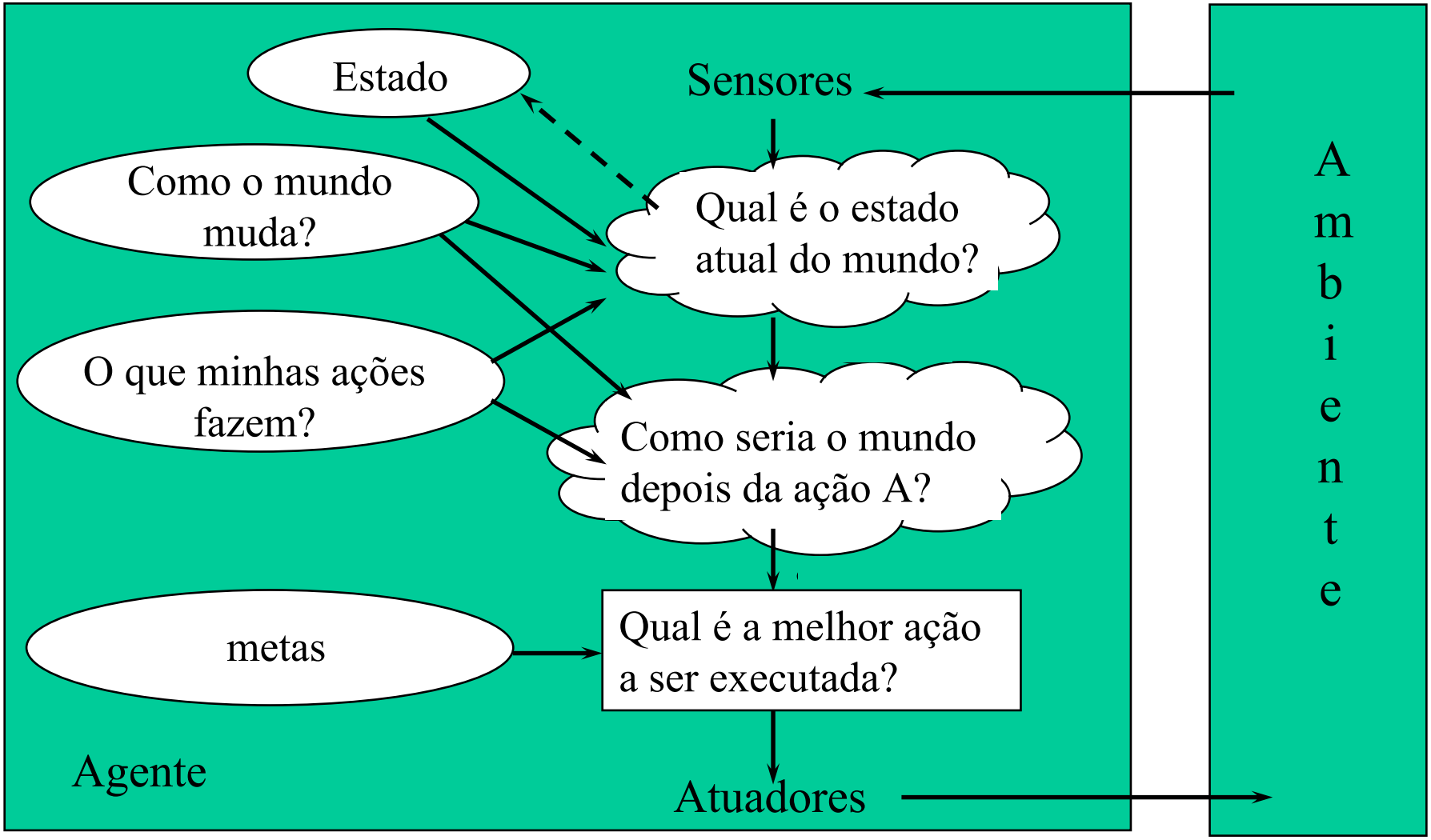
action ← AÇÃO-DA-REGRA(*regra*)

devolve *ação*

Agente baseado em metas

- o mapeamento entre percepções e ações pode ser muito grande ou pode haver mudanças no ambiente ou na medida de desempenho do agente, nesse caso, um agente reativo não funcionaria
- Agente baseado em metas é mais flexível pois contém o conhecimento explícito necessário para a escolha de ações
- Informação da meta: - *O que acontece se essa ação for executada? Eu me aproximo ou afasta da minha meta*?
- Pode ser simples quando uma única ação realiza a meta; outras vezes pode requerer *busca* e *planejamento* (sub-áreas de IA). Pode envolver projeção de ações para prever se a meta é satisfeita.
- Limitação: consome tempo; o mundo pode mudar enquanto se tenta prever o futuro (raciocínio sobre o futuro).

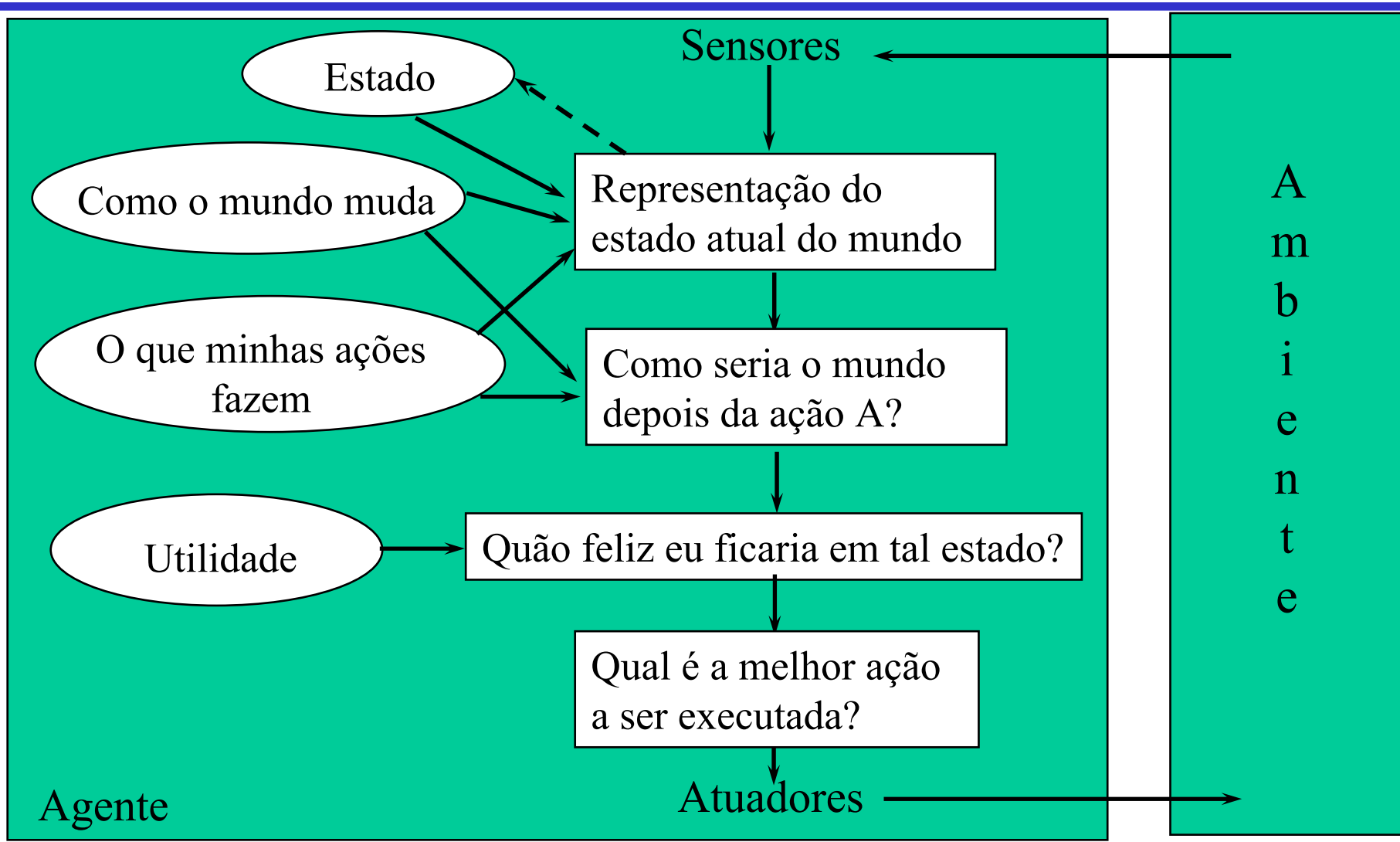
Agente baseado em metas



Agente baseado em utilidade

- Função utilidade: mapeia um estado (ou uma seqüências de estados) a um número real, que descreve o grau de "satisfação" do agente com relação a ação tomada.
- Informa se um estado do mundo é preferível (mais útil) que outros. Avalia metas competitivos e guia a busca.
- Jogos caem nessa categoria
- Gera comportamento de alta qualidade

Agente baseado em utilidade



Agentes com aprendizado

- (Turing, 50): "*construir máquinas com aprendizagem para depois ensiná-las*"

Ambientes

Ambientes

- agentes operam dentro de um ambiente
  - robôs “percebem” o mundo
  - simuladores podem fornecer dados ambientais
- ambientes possuem grande influência sobre o projeto de agentes
- existem ambientes padrões para avaliação de agentes

Propriedades do ambiente

- acessível versus inacessível
- determinístico versus não-determinístico
- estático versus dinâmico
- discreto versus contínuo

Acessível versus inacessível

- se o sistema de sensoriamento do agente dá acesso ao estado completo do ambiente, isto é, todos os aspectos relevantes para a escolha da ação.

Determinístico versus não-determinístico

- Determinístico: se o próximo estado do mundo é completamente determinado pelo estado atual e as ações do agente.
- Do ponto de vista do agente, se o ambiente não é accessível ele também é não-determinístico

Episódico versus não episódico

- Tarefa se divide em sub-tarefas independentes mais simples.

Estático versus dinâmico

- Dinâmico: muda enquanto o agente raciocina (delibera).
- Estático: o agente não precisa se preocupar com a passagem de tempo (alguns tipos de jogos ou problemas de escalonamento).

Discreto versus contínuo

- no. limitado de percepções e ações (definidas e distintas).
  - Xadrez: ambiente discreto. Possue no. fixo de possíveis movimentos em cada jogada e número fixo de posições para cada peça
  - Taxi: ambiente continuo com relação à velocidade e localização

Tipos de ambientes

- diferentes tipos de ambiente requerem programas de agente diferentes
- Agente complexo: ambiente inacessível, dinâmico e contínuo.
- Mundo real: não-determinístico

Tipos de ambientes

Ambiente	Acess.	Determ.	Episoc.	Estat.	Discreto
Xadrez	S	S	N	S	S
Motorista de taxi	N	N	N	N	N
Diagnóstico médico	N	N	N	N	N
Análise de imagens	S	S	S	S	N

Programas de ambiente

- Relação entre agentes e ambientes
- Ambiente simulador (para um ou mais agentes)
- o simulador atualiza o ambiente de acordo com as ações dos agentes e dinâmica do próprio ambiente
- ambiente:
  - estado inicial
  - função de atualização

RUN-ENVIRONMENT

procedure RUN-ENVIRONMENT (*state*, UPDATE-FN, *agents, termination*

inputs *state*, UPDATE-FN, *agents, termination*

repeat

for each *agent* **in** *agents* **do**

PERCEPT[*agent*] ← GET-PERCEPT(*agent, state*)

end

for each *agent* **in** *agents* **do**

ACTION[*agent*] ← PROGRAM[*agent*](PERCEPT[*agent*])

end

state ← UPDATE-FN(*actions, agents, state*)

until *termination*(*state*)

Sumário

- Agente: arquitetura + programa do agente;
- Agente ideal: escolhe a ação que maximiza sua medida de desempenho, dada a seqüência de percepção;
- Agente autônomo: possui experiência própria ao invés de depender do conhecimento pré-codificado sobre o ambiente;
- Projeto (design) do agente depende do tipo de informação disponível e usada no processo de decisão;
- O projeto apropriado depende da descrição PEAS do agente.