CAPÍTULO 2

Agentes inteligentes

Em que discutimos a natureza dos agentes, perfeitos ou não, a diversidade de ambientes e a conseqüente variedade de tipos de agentes.

O Capítulo 1 identificou o conceito de **agentes racionais** como uma questão central para nossa abordagem da inteligência artificial. Neste capítulo, tornaremos essa noção mais concreta. Veremos que o conceito de racionalidade pode ser aplicado a uma ampla variedade de agentes que operam em qualquer ambiente imaginável. Nosso plano neste livro é usar esse conceito para desenvolver um pequeno conjunto de princípios de projeto com a finalidade de construir sistemas de agentes bem-sucedidos – sistemas que possam ser adequadamente chamados **inteligentes**.

Começaremos examinando agentes, ambientes e o acoplamento entre eles. A observação de que alguns agentes se comportam melhor que outros leva naturalmente à idéia de agente racional – um agente que se comporta tão bem quanto possível. A medida da qualidade do comportamento de um agente depende da natureza do ambiente; alguns ambientes são mais difíceis que outros. Apresentaremos uma divisão geral dos ambientes em categorias e mostraremos como as propriedades de um ambiente influenciam o projeto de agentes adequados para esse ambiente. Descreveremos vários projetos de "esqueletos" básicos de agentes que serão utilizados no restante do livro.

2.1 Agentes e ambientes

AMBIENTE
SENSOR
ATUADOR

Um **agente** é tudo o que pode ser considerado capaz de perceber seu **ambiente** por meio de **sensores** e de agir sobre esse ambiente por intermédio de **atuadores**. Essa idéia simples é ilustrada na Figura 2.1. Um agente humano tem olhos, ouvidos e outros órgãos como sensores, e tem mãos, pernas, boca e outras partes do corpo que servem como atuadores. Um agente robótico poderia ter câmeras e detectores da faixa de infravermelho funcionando como sensores e vários motores como atuadores. Um

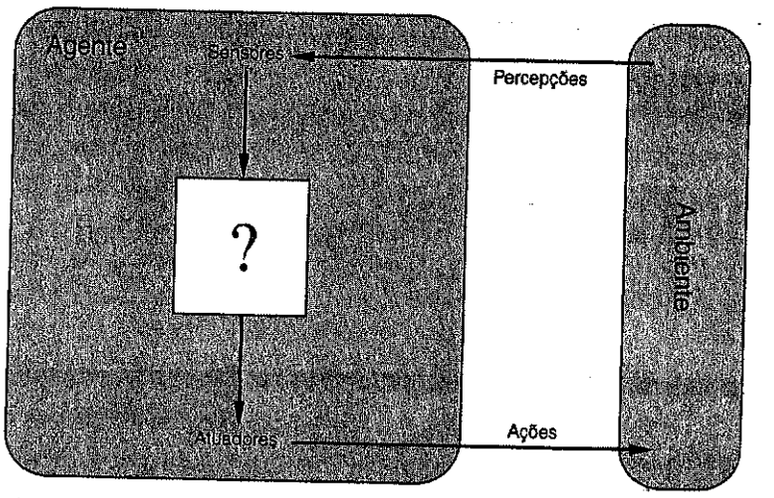

Figura 2.1 Agentes interagem com ambientes por meio de sensores e atuadores.

agente de software recebe seqüências de teclas digitadas, conteúdo de arquivos e pacotes de rede como entradas sensórias e atua sobre o ambiente exibindo algo na tela, gravando arquivos e enviando pacotes de rede. Faremos a suposição geral de que todo agente pode perceber suas próprias ações (mas nem sempre os efeitos).

PERCEPÇÃO

SEQÜÊNCIA DE PERCEPÇÕES

Usamos o termo **percepção** para fazer referência às entradas perceptivas do agente em qualquer momento dado. A **seqüência de percepções** do agente é a história completa de tudo que o agente já percebeu. Em geral, *a escolha de ação de um agente em qualquer instante dado pode depender da seqüência inteira de percepções observadas até o momento*. Se pudermos especificar a escolha de ação do agente para toda seqüência de percepções possível, então teremos dito quase tudo o que existe a dizer sobre o agente. Em termos matemáticos, afirmamos que o comportamento do agente é descrito pela **função de agente** que mapeia qualquer seqüência de percepções específica para uma ação.

FUNÇÃO DE AGENTE

Podemos imaginar a *tabulação* da função de agente que descreve qualquer agente dado; para a maioria dos agentes, o resultado seria uma tabela muito grande – na verdade infinita, a menos que seja definido um limite sobre o comprimento das seqüências de percepções que queremos considerar. Dado um agente para a realização de experimentos, podemos, em princípio, construir essa tabela tentando todas as seqüências de percepções e registrando as ações que o agente executa em resposta.[1] É claro que a tabela é uma caracterização *externa* do agente. *Internamente*, a função de agente para um agente artificial será implementada por um **programa de agente**. É importante manter essas duas idéias distintas. A função de agente é uma descrição matemática abstrata; o programa de agente é uma implementação concreta, relacionada à arquitetura do agente.

PROGRAMA DE AGENTE

Para ilustrar essas idéias, usaremos um exemplo muito simples – o mundo de aspirador de pó ilustrado na Figura 2.2. Esse mundo é tão simples que podemos descrever tudo o que acontece; ele também é um mundo inventado e, portanto, podemos criar muitas variações. Esse mundo particular tem apenas dois locais: os quadrados *A* e *B*. O agente aspirador de pó percebe em que quadrado está e se existe sujeira no quadrado. Ele pode optar por mover-se para a esquerda, mover-se para a direita, aspirar a sujeira ou não fazer nada. Uma função de agente muito simples é: se o quadrado atual estiver sujo, então aspirar, caso contrário mover-se para o outro quadrado. Uma tabulação parcial da função

1. Se o agente utilizasse alguma aleatoriedade para escolher suas ações, teríamos de experimentar cada seqüência muitas vezes para identificar a probabilidade de cada ação. Talvez alguém considere a atuação aleatória bastante tola, mas veremos mais adiante neste capítulo que ela pode ser muito inteligente.

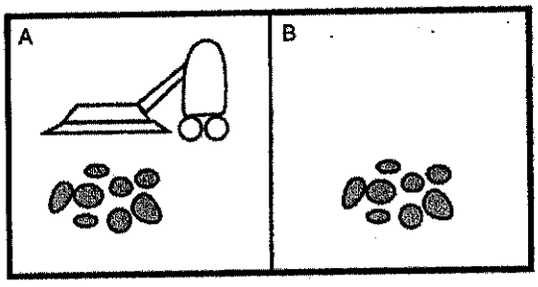

Figura 2.2 Um mundo de aspirador de pó com apenas dois locais.

Seqüência de percepções	Ação
[*A, Limpo*]	*Direita*
[*A, Sujo*]	*Aspirar*
[*B, Limpo*]	*Esquerda*
[*B, Sujo*]	*Aspirar*
[*A, Limpo*], [*A, Limpo*]	*Direita*
[*A, Limpo*], [*A, Sujo*]	*Aspirar*
⋮	⋮
[*A, Limpo*], [*A, Limpo*], [*A, Limpo*]	*Direita*
[*A, Limpo*], [*A, Limpo*], [*A, Sujo*]	*Aspirar*
⋮	⋮

Figura 2.3 Tabulação parcial de uma função de agente simples correspondente ao mundo de aspirador de pó mostrado na Figura 2.2.

desse agente é mostrada na Figura 2.3. Um programa de agente simples para essa função de agente será apresentado mais adiante no capítulo, na Figura 2.8.

Examinando a Figura 2.3, vemos diversos agentes do mundo de aspirador de pó que podem ser definidos, simplesmente preenchendo-se de várias maneiras a coluna da direita. Então, a pergunta óbvia é: *Qual é a maneira correta de preencher a tabela?* Em outras palavras, o que torna um agente bom ou ruim, inteligente ou estúpido? Responderemos a essas perguntas na próxima seção.

Antes de fecharmos esta seção, observaremos que a noção de um agente deve ser vista como uma ferramenta para analisar sistemas, não como uma caracterização absoluta que divide o mundo em agentes e não-agentes. Poderíamos visualizar uma calculadora portátil como um agente que escolhe a ação de exibir "4" ao receber a seqüência de percepções "2 + 2 = ", mas tal análise dificilmente ajudaria nossa compreensão da calculadora.

2.2 Bom comportamento: o conceito de racionalidade

AGENTE RACIONAL

Um **agente racional** é aquele que faz tudo certo – em termos conceituais, toda entrada na tabela correspondente à função de agente é preenchida de forma correta. É óbvio que fazer tudo certo é melhor do que fazer tudo errado; porém, o que significa fazer tudo certo? Como uma primeira abordagem, diremos que a ação certa é aquela que fará o agente obter maior sucesso. Então, precisaremos de algum método para medir o sucesso. Juntamente com a descrição do ambiente e dos sensores e atuadores do agente, esse método fornecerá uma especificação completa da tarefa que o agente deve empreender. Com isso, poderemos definir com maior precisão o que significa ser racional.

Medidas de desempenho

MEDIDA DE DESEMPENHO

Uma **medida de desempenho** encarna o critério para se medir o sucesso do comportamento do agente. Quando um agente é inserido em um ambiente, ele gera uma seqüência de ações, de acordo com as percepções que recebe. Essa seqüência de ações faz o ambiente passar por uma seqüência de estados. Se a seqüência é desejável, isso quer dizer que o agente funcionou bem. Evidentemente, não existe uma medida fixa apropriada para todos os agentes. Poderíamos pedir ao agente uma opinião subjetiva do quanto ele está feliz com seu próprio desempenho, mas alguns agentes seriam incapazes de responder e outros iludiriam a si mesmos.[2] Portanto, insistiremos em uma medida objetiva do desempenho, em geral uma que seja imposta pelo projetista que estiver construindo o agente.

Considere o agente aspirador de pó da seção anterior. Poderíamos propor medir o desempenho pela quantidade de sujeira limpa em um único turno de oito horas. É claro que, no caso de um agente racional, você obtém aquilo que solicita. Um agente racional pode maximizar essa medida de desempenho limpando a sujeira e, em seguida, despejando tudo no chão, depois limpando novamente e assim por diante. Uma medida de desempenho mais apropriada recompensaria o agente por deixar o chão limpo. Por exemplo, ele poderia ser recompensado por cada quadrado limpo em cada período (talvez com uma penalidade pela eletricidade consumida e pelo ruído gerado). *Como regra geral, é melhor projetar medidas de desempenho de acordo com o resultado realmente desejado no ambiente, em vez de criá-las de acordo com o comportamento esperado do agente.*

A seleção de uma medida de desempenho nem sempre é fácil. Por exemplo, a noção de "chão limpo" no parágrafo anterior se baseia na limpeza média ao longo do tempo. Ainda assim, a mesma limpeza média pode ser alcançada por dois agentes diferentes, um dos quais faz o trabalho medíocre o tempo todo, enquanto o outro limpa energicamente, mas faz longas pausas. A estratégia preferível pode parecer um detalhe secundário da ciência da zeladoria, mas de fato é uma profunda questão filosófica com extensas implicações. O que é melhor – uma vida aventureira, cheia de altos e baixos, ou uma existência segura, porém monótona? O que é melhor – uma economia em que todos vivem em uma pobreza moderada ou aquela em que alguns vivem em plena riqueza enquanto outros são muito pobres? Deixaremos essas perguntas como um exercício para o leitor diligente.

Racionalidade

A definição do que é racional em qualquer instante dado depende de quatro fatores:

- A medida de desempenho que define o critério de sucesso.
- O conhecimento anterior que o agente tem do ambiente.
- As ações que o agente pode executar.
- A seqüência de percepções do agente até o momento.

DEFINIÇÃO DE UM AGENTE RACIONAL

Isso conduz a uma **definição de um agente racional**:

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

2. Em particular, os agentes humanos são notórios pelas "uvas verdes" – acreditar que na realidade não querem algo depois de tentarem sem sucesso obtê-lo, como em: "Tudo bem, não importa, eu não queria mesmo aquele ridículo prêmio Nobel."

Considere o agente aspirador de pó simples que limpa um quadrado se ele estiver sujo e passa para o outro quadrado se o primeiro não estiver sujo; essa é a função do agente tabulada na Figura 2.3. Esse é um agente racional? Depende! Primeiro, precisamos dizer o que é a medida de desempenho, o que se conhece sobre o ambiente e quais são os sensores e atuadores que o agente tem. Vamos supor que:

- A medida de desempenho ofereça o prêmio de um ponto para cada quadrado limpo em cada período de tempo, ao longo de uma "duração" de 1.000 períodos de tempo.
- A "geografia" do ambiente é conhecida *a priori* (Figura 2.2), mas a distribuição da sujeira e a posição inicial do agente não são previamente conhecidas. Quadrados limpos permanecem limpos e a aspiração limpa o quadrado atual. As ações *Esquerda* e *Direita* movem o agente para a esquerda e para a direita, exceto quando isso leva o agente para fora do ambiente; nesse caso, o agente permanece onde está.
- As únicas ações disponíveis são *Esquerda*, *Direita*, *Aspirar* e *NoOp* (não fazer nada).
- O agente percebe corretamente sua posição e se essa posição contém sujeira.

Afirmamos que, *sob essas circunstâncias*, o agente é de fato racional; espera-se que seu desempenho seja pelo menos tão elevado quanto o de qualquer outro agente. O Exercício 2.4 lhe pede para provar esse fato.

Podemos ver facilmente que o mesmo agente seria irracional sob circunstâncias diferentes. Por exemplo, uma vez que toda a sujeira seja limpa, ele oscilará desnecessariamente de um lado para outro; se a medida de desempenho incluir uma penalidade de um ponto para cada movimento à esquerda ou à direita, o agente ficará em má situação. Um agente melhor para esse caso não faria nada se tivesse certeza de que todos os quadrados estão limpos. Se quadrados limpos puderem ficar sujos de novo, o agente deve ocasionalmente verificar e voltar a limpá-los, se necessário. Se a geografia do ambiente for desconhecida, o agente precisará explorá-la, em vez de se fixar nos quadrados A e B. O Exercício 2.4 lhe pede para projetar agentes relativos a esses casos.

Onisciência, aprendizado e autonomia

ONISCIÊNCIA

Precisamos ter o cuidado de distinguir entre racionalidade e **onisciência**. Um agente onisciente sabe o resultado *real* de suas ações e pode agir de acordo com ele; porém, a onisciência é impossível na realidade. Considere o exemplo a seguir: estou caminhando nos Champs Elysées e de repente vejo um velho amigo do ouro lado da rua. Não existe nenhum tráfego perto e não tenho nenhum outro compromisso; assim, sendo racional, começo a atravessar a rua. Enquanto isso, a 10.000 metros de altura, a porta do compartimento de carga se solta de um avião[3] e, antes de chegar ao outro lado da rua, sou atingido. Foi irracional atravessar a rua? É improvável que a notícia de minha morte fosse "idiota tenta cruzar rua".

Esse exemplo mostra que racionalidade não é o mesmo que perfeição. A racionalidade maximiza o desempenho *esperado*, enquanto a perfeição maximiza o desempenho *real*. Fugir à exigência de perfeição não é apenas uma questão de ser justo com os agentes. Se esperarmos que um agente realize aquela que virá a ser a melhor ação após o fato, será impossível projetar um agente para satisfazer a essa especificação – a menos que melhoremos o desempenho de bolas de cristal ou máquinas do tempo.

3. Veja N. Henderson, "New door latches urged for Boeing 747 jumbo jets", *Washington Post*, 24 de agosto de 1989.

Portanto, nossa definição de racionalidade não exige onisciência, porque a escolha racional só depende da seqüência de percepções *até o momento*. Também devemos assegurar que não permitimos que o agente se engaje sem querer em atividades decididamente pouco inteligentes. Por exemplo, se um agente não olhar para os dois lados antes de atravessar uma estrada movimentada, sua seqüência de percepções não o informará de que existe um grande caminhão se aproximando em alta velocidade. Nossa definição de racionalidade afirmaria que agora é correto atravessar a estrada? Longe disso! Primeiro, não seria racional atravessar a estrada dada essa seqüência de percepções pouco informativa: o risco de acidente resultante de atravessar a estrada sem olhar para os lados é muito grande. Em segundo lugar, um agente racional deve escolher a ação "olhar" antes de iniciar a travessia, porque olhar ajuda a maximizar o desempenho esperado. A realização de ações *com a finalidade de modificar percepções futuras* – às vezes chamada **coleta de informações** – é uma parte importante da racionalidade e é abordada em profundidade no Capítulo 16. Um segundo exemplo de coleta de informações é dado pela **exploração** que tem de ser empreendida por um agente aspirador de pó em um ambiente inicialmente desconhecido.

COLETA DE INFORMAÇÕES

EXPLORAÇÃO

Nossa definição exige um agente racional não apenas para coletar informações, mas também para **aprender** tanto quanto possível do que ele percebe. A configuração inicial do agente poderia refletir algum conhecimento anterior do ambiente mas, à medida que o agente ganha experiência, isso pode ser modificado e ampliado. Existem casos extremos em que o ambiente é completamente conhecido *a priori*. Em tais casos, o agente não precisa perceber ou aprender; ele simplesmente age de forma correta. É claro que tais agentes são muito frágeis. Considere o humilde besouro de esterco. Depois de cavar seu ninho e depositar os ovos, ele busca uma bola de esterco em um monte próximo para fechar a entrada. Se, *durante o percurso*, a bola de esterco for removida de suas garras, o besouro seguirá em frente e imitará o fechamento do ninho com a bola de esterco inexistente, sem notar que ela foi retirada. A evolução construiu uma suposição sobre o comportamento do besouro e, quando essa hipótese é violada, resulta um comportamento malsucedido. A vespa sphex é um pouco mais inteligente. A fêmea da sphex cava uma cova, sai, pica uma lagarta e a arrasta até a borda da cova, entra novamente na cova para verificar se tudo está bem, arrasta a lagarta para dentro e deposita seus ovos. A lagarta servirá como alimento quando os ovos eclodirem. Até aqui tudo bem, mas se um entomologista afastar a lagarta algumas polegadas enquanto a fêmea estiver fazendo a verificação, ela voltará à etapa de "arrastar" de seu plano e continuará o plano sem modificação, mesmo depois de dezenas de intervenções de afastamento de lagartas. A sphex é incapaz de aprender que seu plano inato está falhando, e portanto não o modificará.

APRENDIZAGEM

Os agentes bem-sucedidos dividem a tarefa de calcular a função do agente em três períodos distintos: quando o agente está sendo projetado, uma parte do cálculo é feita por seus projetistas; quando está deliberando sobre sua próxima ação, o agente realiza mais cálculos e, à medida que aprende a partir da experiência, ele efetua ainda mais cálculos para decidir como modificar seu comportamento.

Quando um agente se baseia no conhecimento anterior de seu projetista e não em suas próprias percepções, dizemos que o agente não tem **autonomia**. Um agente racional deve ser autônomo – ele deve aprender o que puder para compensar um conhecimento prévio parcial ou incorreto. Por exemplo, um agente aspirador de pó que aprende a prever onde e quando aparecerá mais sujeira funcionará melhor que um agente incapaz de fazer essa previsão. Na prática, raramente se exige autonomia completa desde o início: quando o agente tem pouca ou nenhuma experiência, ele deve agir ao acaso, a menos que o projetista tenha dado a ele alguma assistência. Então, da mesma forma que a evolução fornece aos animais reflexos internos suficientes para que eles possam sobreviver pelo tempo necessário para aprenderem por si mesmos, seria razoável fornecer a um agente de inteligência artificial algum conhecimento inicial, bem como habilidade para aprender. Depois de adquirir experiência suficiente sobre seu ambiente, o comportamento de um agente racional pode se

AUTONOMIA

tornar efetivamente *independente* de seu conhecimento anterior. Em conseqüência disso, a incorporação do aprendizado permite projetar um único agente racional que terá sucesso em uma ampla variedade de ambientes.

2.3 A natureza dos ambientes

AMBIENTES DE TAREFAS

Agora que temos uma definição de racionalidade, estamos quase prontos para pensar em construir agentes racionais. Porém, primeiro devemos pensar em **ambientes de tarefas**, que são essencialmente os "problemas" para os quais os agentes racionais são as "soluções". Começamos mostrando como especificar um ambiente de tarefa, ilustrando o processo com vários exemplos. Em seguida, mostramos que há vários tipos de ambientes de tarefas. O tipo de ambiente de tarefa afeta diretamente o projeto apropriado para o programa do agente.

Especificando o ambiente de tarefa

Em nossa discussão sobre a racionalidade do simples agente aspirador de pó, tivemos de especificar a medida de desempenho, o ambiente, e os atuadores e sensores do agente. Agruparemos todos esses itens sob o título **ambiente de tarefa**. Para os leitores que gostam de acrônimos, chamaremos essa descrição de **PEAS** (Performance, Environment, Actuators, Sensors – desempenho, ambiente, atuadores, sensores). Ao projetar um agente, a primeira etapa deve ser sempre especificar o ambiente de tarefa de forma tão completa quanto possível.

PEAS

O mundo de aspirador de pó foi um exemplo simples; vamos considerar um problema mais complexo: um motorista de táxi automatizado. Utilizaremos esse exemplo em todo o restante do capítulo. Devemos destacar, antes que o leitor fique alarmado, que um táxi totalmente automatizado no momento está um pouco além da capacidade da tecnologia atual. (Veja na página 28 uma descrição de um robô motorista ou então examine os anais recentes das conferências sobre sistemas de transporte inteligentes.) A tarefa de dirigir é extremamente *aberta*. Não existe nenhum limite para as novas combinações de circunstâncias que podem surgir – outra razão para termos escolhido essa tarefa como foco de discussão. A Figura 2.4 resume a descrição do PEAS para o ambiente de tarefa do táxi. Descreveremos cada elemento com mais detalhes nos próximos parágrafos.

Primeiro, que **medida de desempenho** gostaríamos que nosso motorista automatizado tivesse como objetivo? As qualidades desejáveis incluem chegar ao destino correto, minimizar o consumo de combustível e desgaste, minimizar o tempo e/ou o custo de viagem, minimizar violações às leis de trânsito e as perturbações a outros motoristas, maximizar a segurança e o conforto dos passageiros e maximizar os lucros. É óbvio que alguns desses objetivos serão conflitantes; portanto, haverá compromissos.

Em seguida, qual é o **ambiente** de direção que o táxi enfrentará? Qualquer motorista de táxi deve lidar com diversos tipos de estradas, variando desde estradas rurais e avenidas urbanas até rodovias com 12 pistas. As estradas contêm outros tipos de tráfego, pedestres, animais perdidos, trabalhadores na pista, policiamento, poças e buracos. O táxi também deve interagir com passageiros potenciais e reais. Existem ainda algumas escolhas opcionais. O táxi poderia precisar operar no sul da Califórnia, onde a neve raramente é um problema, ou no Alasca, onde ela raramente não é um problema. Ele sempre poderia estar dirigindo no lado direito da pista ou talvez quiséssemos que ele fosse flexível o bastante para dirigir no lado esquerdo quando estivesse na Inglaterra ou no Japão. É óbvio que, quanto mais restrito o ambiente, mais fácil se torna o problema de projetar.

Os **atuadores** disponíveis para um táxi automatizado serão aproximadamente os mesmos disponíveis para um motorista humano: controle sobre o motor através do acelerador e controle sobre a dire-

Tipo de agente	Medida de desempenho	Ambiente	Atuadores	Sensores
Motorista de táxi	Viagem segura, rápida, dentro da lei, confortável, maximizar lucros	Estradas, outros tipos de tráfego, pedestres, clientes	Direção, acelerador, freio, sinal, buzina, visor	Câmeras, sonar, velocímetro, GPS, hodômetro, acelerômetro, sensores do motor, teclado

Figura 2.4 Descrição de PEAS do ambiente de tarefa para um táxi automatizado.

ção e a frenagem. Além disso, ele precisará da saída para uma tela de exibição ou um sintetizador de voz para se comunicar com os passageiros, e talvez algum meio para se comunicar com outros veículos, de forma educada ou não.

Para alcançar seus objetivos no ambiente de direção, o táxi precisará saber onde está, o que mais existe na estrada e qual a sua velocidade. Seus **sensores** básicos devem então incluir uma ou mais câmeras de TV controláveis, o velocímetro e o hodômetro. Para controlar o veículo de forma correta, especialmente em curvas, o táxi deve ter um acelerômetro; ele também terá de conhecer o estado mecânico do veículo e portanto precisará do conjunto habitual de sensores do motor e do sistema elétrico. O táxi automatizado poderia ter instrumentos não-disponíveis para o motorista humano médio: um sistema de posicionamento global por satélite (GPS – global positioning system) lhe dará informações precisas sobre sua posição com relação a um mapa eletrônico, e sensores de infravermelho ou de sonar para detectar distâncias até outros carros e obstáculos. Finalmente, ele precisará de um teclado ou microfone para que o passageiro possa solicitar um destino.

Na Figura 2.5, esboçamos os elementos básicos do PEAS para diversos tipos de agentes. Exemplos adicionais aparecem no Exercício 2.5. Talvez seja surpresa para alguns leitores o fato de incluirmos em nossa lista de tipos de agentes alguns programas que operam no ambiente completamente artificial definido pela entrada no teclado e pela saída de caracteres em uma tela. Alguém poderia dizer: "Certamente, esse não é um ambiente real, é?" De fato, o que importa não é a distinção entre ambientes "reais" e "artificiais", mas a complexidade do relacionamento entre o comportamento do agente, a seqüência de percepções gerada pelo ambiente e a medida de desempenho. Alguns ambientes "reais"

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

Figura 2.5 Exemplos de tipos de agentes e suas descrições de PEAS.

na realidade são bastante simples. Por exemplo, um robô projetado para inspecionar peças à medida que elas chegam em uma correia transportadora pode fazer uso de uma série de suposições simplificadoras: que a iluminação será sempre perfeita, que os únicos itens na correia transportadora serão peças de um tipo que ele conhece e que sempre haverá apenas duas ações (aceitar ou rejeitar).

AGENTES DE SOFTWARE
SOFTBOTS

Em contraste, existem alguns **agentes de software** (ou robôs de software, ou ainda **softbots**) em ambientes ricos e ilimitados. Imagine um softbot projetado para pilotar um simulador de vôo semelhante a um grande avião comercial. O simulador é um ambiente muito detalhado e complexo que inclui outras aeronaves e operações no solo, e o agente de software deve escolher uma dentre várias ações em tempo real. Ou, então, imagine um softbot projetado para vasculhar fontes de notícias da Internet e mostrar os itens interessantes a seus clientes. Para funcionar bem, ele precisará de algumas habilidades de processamento de linguagem natural, precisará aprender o que interessa a cada cliente e terá de mudar seus planos dinamicamente – por exemplo, quando a conexão para uma fonte de notícias cair ou quando uma nova fonte estiver on-line. A Internet é um ambiente cuja complexidade rivaliza com a do mundo físico e cujos habitantes incluem muitos agentes artificiais.

Propriedades de ambientes de tarefas

A variedade de ambientes de tarefas que podem surgir em IA é sem dúvida vasta. Entretanto, podemos identificar um número bastante reduzido de dimensões ao longo das quais os ambientes de tarefas podem ser divididos em categorias. Em grande parte, essas dimensões determinam o projeto apropriado de agentes e a aplicabilidade de cada uma das principais famílias de técnicas de implementação de agentes. Primeiro, listamos as dimensões, depois analisamos vários ambientes de tarefas para ilustrar as idéias. Aqui, as definições são informais; os capítulos posteriores fornecerão enunciados e exemplos mais precisos de cada tipo de ambiente.

COMPLETAMENTE OBSERVÁVEL

- **Completamente observável** *versus* **parcialmente observável.**
  Se os sensores de um agente permitem acesso ao estado completo do ambiente em cada instante, dizemos que o ambiente de tarefa é completamente observável.[4] Um ambiente de tarefa é de fato completamente observável se os sensores detectam todos os aspectos que são *relevantes* para a escolha da ação; por sua vez, a relevância depende da medida de desempenho. Ambientes completamente observáveis são convenientes porque o agente não precisa manter qualquer estado interno para controlar o mundo. Um ambiente poderia ser parcialmente observável devido ao ruído e a sensores imprecisos ou porque partes do estado estão simplesmente ausentes nos dados do sensor – por exemplo, um agente aspirador de pó com apenas um sensor de sujeira local não pode saber se há sujeira em outros quadrados, e um táxi automatizado não pode saber o que outros motoristas estão pensando.

DETERMINÍSTICO
ESTOCÁSTICO

- **Determinístico** *versus* **estocástico.**
  Se o próximo estado do ambiente é completamente determinado pelo estado atual e pela ação executada pelo agente, dizemos que o ambiente é determinístico; caso contrário, ele é estocástico. Em princípio, um agente não precisa se preocupar com a incerteza em um ambiente completamente observável e determinístico. Porém, se o ambiente for parcialmente observável, ele poderá *parecer* estocástico. Isso é particularmente verdadeiro se o ambiente é complexo, tornando difícil controlar todos os aspectos não-observados. Desse modo, freqüentemente é me-

4. A primeira edição deste livro usou os termos **acessível** e **inacessível** em lugar de **completamente** e **parcialmente observáveis**, **não-determinístico** em vez de **estocástico** e **não-episódico** em vez de **seqüencial**. A nova terminologia é mais consistente com o uso estabelecido.

lhor considerar um ambiente determinístico ou estocástico *do ponto de vista do agente*. O motorista de táxi é claramente estocástico nesse sentido, porque nunca se pode prever o comportamento do tráfego com exatidão; além disso, pode ocorrer o estouro de pneu e a falha de um motor sem aviso prévio. O mundo de aspirador de pó que descrevemos é determinístico, mas as variações podem incluir elementos estocásticos como o aparecimento de sujeira ao acaso e um mecanismo de sucção não-confiável (Exercício 2.12). Se o ambiente é determinístico exceto pelas ações de outros agentes, dizemos que o ambiente é **estratégico**.

ESTRATÉGICO

EPISÓDICO
SEQÜENCIAL

- **Episódico** *versus* **seqüencial**.[5]

Em um ambiente de tarefa episódico, a experiência do agente é dividida em episódios atômicos. Cada episódio consiste na percepção do agente, e depois na execução de uma única ação. É crucial que o episódio seguinte não dependa das ações executadas em episódios anteriores. Em ambientes episódicos, a escolha da ação em cada episódio só depende do próprio episódio. Muitas tarefas de classificação são episódicas. Por exemplo, um agente que tem de localizar peças defeituosas em uma linha de montagem baseia cada decisão na peça atual, independente das decisões anteriores; além disso, a decisão atual não afeta o fato de a próxima peça estar ou não com defeito. Por outro lado, em ambientes seqüenciais, a decisão atual poderia afetar todas as decisões futuras. Jogar xadrez e dirigir um táxi são seqüenciais: em ambos os casos, ações em curto prazo podem ter conseqüências a longo prazo. Ambientes episódicos são muito mais simples que ambientes seqüenciais, porque o agente não precisa pensar à frente.

ESTÁTICO
DINÂMICO

- **Estático** *versus* **dinâmico**.

Se o ambiente puder se alterar enquanto um agente está deliberando, dizemos que o ambiente é dinâmico para esse agente; caso contrário, ele é estático. Ambientes estáticos são fáceis de manipular, porque o agente não precisa continuar a observar o mundo enquanto está decidindo sobre a realização de uma ação, nem precisa se preocupar com a passagem do tempo. Por outro lado, ambientes dinâmicos estão continuamente perguntando ao agente o que ele deseja fazer; se ele ainda não tiver se decidido, isso será considerado a decisão de não fazer nada. Se o próprio ambiente não mudar com a passagem do tempo, mas o nível de desempenho do agente se alterar, diremos que o ambiente é **semidinâmico**. O ambiente em que se dirige um táxi é claramente dinâmico: os outros carros e o próprio táxi continuam a se mover enquanto o algoritmo de direção hesita sobre o que fazer em seguida. O jogo de xadrez, quando jogado com a contagem do tempo, é semidinâmico. O jogo de palavras cruzadas é estático.

SEMIDINÂMICO

DISCRETO
CONTÍNUO

- **Discreto** *versus* **contínuo**.

A distinção entre discreto e contínuo pode se aplicar ao *estado* do ambiente, ao modo como o *tempo* é tratado, e ainda às *percepções* e *ações* do agente. Por exemplo, um ambiente de estados discretos como um jogo de xadrez tem um número finito de estados distintos. O xadrez também tem um conjunto discreto de percepções e ações. Dirigir um táxi é um problema de estado contínuo e tempo contínuo: a velocidade e a posição do táxi e dos outros veículos passam por um intervalo de valores contínuos e fazem isso suavemente ao longo do tempo. As ações de dirigir um táxi também são contínuas (ângulos de rotação do volante etc.). A entrada proveniente de câmeras digitais é discreta, em termos estritos, mas em geral é tratada como a representação de intensidades e posições que variam continuamente.

AGENTE ÚNICO
MULTIAGENTE

- **Agente único** *versus* **multiagente**.

A distinção entre ambientes de agente único e de multiagente pode parecer bastante simples. Por exemplo, um agente que resolve um jogo de palavras cruzadas sozinho está claramente em

5. A palavra "seqüencial" também é usada em ciência da computação como antônimo de "paralelo". Os dois significados não têm muita correlação.

um ambiente de agente único, enquanto um agente que joga xadrez está em um ambiente de dois agentes. Porém, existem algumas questões sutis. Primeiro, descrevemos como uma entidade *pode* ser visualizada como um agente, mas não explicamos que entidades *devem* ser visualizadas como agentes. Um agente A (por exemplo, o motorista de táxi) tem de tratar um objeto B (outro veículo) como um agente ou ele pode ser tratado apenas como um objeto que tem um comportamento estocástico, análogo ao das ondas do mar ou das folhas espalhadas pelo vento? A distinção fundamental é saber se o comportamento de B é ou não mais bem descrito como a maximização de uma medida de desempenho cujo valor depende do comportamento do agente A. Por exemplo, em xadrez, a entidade oponente B está tentando maximizar sua medida de desempenho que, pelas regras de xadrez, minimiza a medida de desempenho do agente A. Desse modo, o jogo de xadrez é um ambiente de multiagente **competitivo**. Por outro lado, no ambiente de direção de um táxi, evitar colisões maximiza a medida de desempenho de todos os agentes; assim, esse é um ambiente de multiagente parcialmente **cooperativo**. Ele também é parcialmente competitivo porque, por exemplo, apenas um carro pode ocupar um espaço no estacionamento. Os problemas de projeto de agentes que surgem em ambientes de multiagentes muitas vezes são bem diferentes dos que surgem em ambientes de um único agente; por exemplo, a **comunicação** com freqüência emerge como um comportamento racional em ambientes de multiagentes; em alguns ambientes competitivos parcialmente observáveis, o **comportamento estocástico** é racional porque evita as armadilhas da previsibilidade.

COMPETITIVO

COOPERATIVO

Como se poderia esperar, o caso mais difícil é *parcialmente observável, estocástico, seqüencial, dinâmico, contínuo* e *multiagente*. Também concluímos que a maioria das situações reais é tão complexa que o fato de serem *realmente* determinísticas é um ponto discutível. Para finalidades práticas, elas devem ser tratadas como estocásticas. Dirigir um táxi é difícil em todos esses sentidos.

A Figura 2.6 lista as propriedades de vários ambientes familiares. Observe que as respostas nem sempre são definitivas. Por exemplo, listamos xadrez como completamente observável; no sentido exato, isso é falso, porque certas regras sobre o roque, captura *en passant* e empates por repetição de movimentos exigem a memorização de alguns fatos sobre o histórico do jogo que não são observáveis como parte do estado do tabuleiro. Essas exceções à possibilidade de observação são sem dúvida de menor importância em comparação àquelas encontradas pelo motorista de táxi, pelo instrutor de inglês ou pelo sistema de diagnóstico médico.

Algumas outras respostas na tabela dependem da forma como o ambiente de tarefa é definido. Listamos a tarefa de diagnóstico médico como uma tarefa de agente único porque o processo de doença em um paciente não poderia ser modelado de modo proveitoso como um agente; porém, um sistema de diagnóstico médico também poderia ter de lidar com pacientes obstinados e pessoal cético, e assim o ambiente poderia ter um aspecto multiagente. Além disso, o diagnóstico médico é episódico se a tarefa for concebida como a seleção de um diagnóstico dada uma lista de sintomas; o problema será seqüencial se a tarefa puder incluir a proposição de uma série de testes, a avaliação do progresso durante o tratamento e assim por diante. Também há muitos ambientes episódicos em níveis mais altos que as ações individuais do agente. Por exemplo, um torneio de xadrez consiste em uma seqüência de jogos; cada jogo é um episódio, porque (em geral) a contribuição dos movimentos em um jogo para o desempenho global do agente não é afetada pelos movimentos de seu jogo anterior. Por outro lado, a tomada de decisões em um único jogo certamente é seqüencial.

O repositório de código associado a este livro (aima.cs.berkeley.edu) inclui implementações de vários ambientes, juntamente com um simulador de ambiente de uso geral que coloca um ou mais agentes em um ambiente simulado, observa seu comportamento com o passar do tempo e os avalia de acordo com uma determinada medida de desempenho. Com freqüência, tais experimentos são executados não para um único ambiente, mas para muitos ambientes extraídos de uma **classe de ambientes**. Por exem-

CLASSE DE AMBIENTES

Ambiente de tarefa	Observável	Determinístico	Episódico	Estático	Discreto	Agentes
Jogo de palavras cruzadas	Completamente	Determinístico	Seqüencial	Estático	Discreto	Único
Xadrez com um relógio	Completamente	Estratégico	Seqüencial	Semi	Discreto	Multi
Pôquer	Parcialmente	Estratégico	Seqüencial	Estático	Discreto	Multi
Gamão	Completamente	Estocástico	Seqüencial	Estático	Discreto	Multi
Direção de táxi	Parcialmente	Estocástico	Seqüencial	Dinâmico	Contínuo	Multi
Diagnóstico médico	Parcialmente	Estocástico	Seqüencial	Dinâmico	Contínuo	Único
Análise de imagens	Completamente	Determinístico	Episódico	Semi	Contínuo	Único
Robô de seleção de peças	Parcialmente	Estocástico	Episódico	Dinâmico	Contínuo	Único
Controlador de refinaria	Parcialmente	Estocástico	Seqüencial	Dinâmico	Contínuo	Único
Instrutor interativo de inglês	Parcialmente	Estocástico	Seqüencial	Dinâmico	Discreto	Multi

Figura 2.6 Exemplos de ambientes de tarefas e suas características.

plo, avaliar um motorista de táxi em tráfego simulado requer a execução de muitas simulações com diferentes condições de tráfego, iluminação e tempo. Se projetássemos o agente para um único cenário, poderíamos tirar proveito de propriedades específicas do caso particular, mas não poderíamos criar um bom projeto para dirigir de maneira geral. Por essa razão, o repositório de código também inclui um **gerador de ambientes** para cada classe de ambientes que seleciona ambientes específicos (com certas variações aleatórias) nos quais seria possível executar o agente. Por exemplo, o gerador de ambientes de aspirador de pó inicializa o padrão de sujeira e a posição do agente de forma aleatória. Então, estamos interessados no desempenho médio do agente sobre a classe de ambientes. Um agente racional para uma dada classe de ambientes maximiza seu desempenho médio. Os Exercícios 2.7 a 2.12 conduzem o leitor pelo processo de desenvolver uma classe de ambientes e de avaliar diversos agentes dentro dessa classe.

GERADOR DE AMBIENTE

2.4 A estrutura de agentes

Até agora fizemos referência aos agentes descrevendo o *comportamento* – a ação executada após qualquer seqüência de percepções específica. Agora, teremos de seguir em frente e examinar o funcionamento interno desses agentes. O trabalho da IA é projetar o **programa de agente** que implementa a função de agente que mapeia percepções em ações. Supomos que esse programa será executado em algum tipo de dispositivo de computação com sensores e atuadores físicos – chamamos esse conjunto de **arquitetura:**

PROGRAMA DE AGENTE

ARQUITETURA

$$agente = arquitetura + programa$$

É óbvio que o programa que escolhermos tem de ser apropriado para a arquitetura. Se o programa recomendar ações como *Caminhar*, é melhor que a arquitetura tenha pernas. A arquitetura pode ser apenas um PC comum ou talvez um carro robótico com diversos computadores, câmeras e outros sensores a bordo. Em geral, a arquitetura torna as percepções dos sensores disponíveis para o programa, executa o programa e alimenta as opções de ação do programa para os atuadores à medida que eles são gerados. A maior parte deste livro trata do projeto de programas de agentes, embora os Capítulos 24 e 25 lidem diretamente com os sensores e atuadores.

Programas de agentes

Os programas de agentes que projetaremos neste livro têm todos a mesma estrutura básica: eles recebem a percepção atual como entrada dos sensores e retornam uma ação para os atuadores.[6] Note a diferença entre o programa de agente, que toma a percepção atual como entrada, e a função de agente, que recebe o histórico de percepções completo. O programa de agente toma apenas a percepção atual como entrada, como nada mais está disponível a partir do ambiente; se as ações do agente dependerem da seqüência de percepções inteira, o agente terá de memorizar as percepções.

Descreveremos os programas de agentes por meio da linguagem de pseudocódigo simples definida no Apêndice B. (O repositório de código on-line contém implementações em linguagens de programação reais.) Por exemplo, a Figura 2.7 mostra um programa de agente bastante trivial que controla a seqüência de percepções, e depois a utiliza para realizar a indexação em uma tabela de ações, a fim de decidir o que fazer. A tabela representa explicitamente a função de agente que o programa de agente incorpora. Para construir um agente racional desse modo, devemos construir uma tabela que contenha a ação apropriada para todas as seqüências de percepções possíveis.

É instrutivo considerar por que a abordagem orientada a tabelas para construção de agentes está condenada ao fracasso. Seja $\mathcal{P}$ o conjunto de percepções possíveis e seja T o tempo de duração do agente (o número total de percepções que ele receberá). A tabela de pesquisa conterá $\sum_{t=1}^{T} |\mathcal{P}|^t$ entradas. Considere o táxi automatizado: a entrada visual de uma única câmera chega à velocidade de aproximadamente 27 megabytes por segundo (30 quadros por segundo, 640 × 480 pixels com 24 bits de informações de cores). Isso nos dá uma tabela de pesquisa com mais de $10^{250.000.000.000}$ entradas para uma hora de direção. Até mesmo a tabela de pesquisa para o xadrez – um minúsculo e bem-comportado fragmento do mundo real – teria pelo menos 10^{150} entradas. O assustador tamanho dessas tabelas (o número de átomos no universo observável é menor que 10^{80}) significa que (a) nenhum agente físico nesse universo terá espaço para armazenar a tabela, (b) o projetista não teria tempo para criar a tabela, (c) nenhum agente poderia sequer apreender todas as entradas de tabelas corretas a partir de sua experiência e (d) mesmo que o ambiente seja simples o bastante para gerar uma tabela de tamanho viável, o projetista ainda não terá nenhuma orientação sobre como inserir as entradas da tabela.

Apesar de tudo isso, o AGENTE-DIRIGIDO-POR-TABELA *faz* o que queremos: implementa a função de agente desejada. O desafio fundamental da IA é descobrir como escrever programas que, na medida do possível, produzam um comportamento racional a partir de uma pequena quantidade

```
função AGENTE-DIRIGIDO-POR-TABELA(percepção) retorna uma ação
  variáveis estáticas: percepções, uma seqüência, inicialmente vazia
        tabela, uma tabela de ações, indexada por seqüências de percepções, de início
completamente especificada

  anexar percepção ao fim de percepções
  ação ← ACESSAR(percepções, tabela)
  retornar ação
```

Figura 2.7 O programa AGENTE-DIRIGIDO-POR-TABELA é invocado para cada nova percepção e retorna uma ação de cada vez. Ele mantém o controle da seqüência de percepções usando sua própria estrutura de dados privada.

6. Existem outras opções para a estrutura do programa de agente; por exemplo, poderíamos fazer os programas dos agentes serem **co-rotinas** que são executadas de forma assíncrona com o ambiente. Cada uma dessas co-rotinas tem uma porta de entrada e uma porta de saída, e consiste em um loop que lê a porta de entrada em busca de percepções e grava ações na porta de saída.

de código, e não a partir de um grande número de entradas de tabelas. Temos muitos exemplos mostrando que isso pode ser feito com sucesso em outras áreas: por exemplo, as enormes tabelas de raízes quadradas usadas por engenheiros e por estudantes antes da década de 1970 foram substituídas por um programa de cinco linhas que corresponde ao método de Newton e é executado em calculadoras eletrônicas. A pergunta é: a IA pode fazer pelo comportamento inteligente em geral o que Newton fez para as raízes quadradas? Acreditamos que a resposta seja sim.

No restante desta seção, descreveremos quatro tipos básicos de programas de agentes que incorporam os princípios subjacentes a quase todos os sistemas inteligentes:

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

Em seguida, explicaremos em termos gerais como converter todos esses tipos básicos em *agentes com aprendizado*.

Agentes reativos simples

AGENTE REATIVO SIMPLES

O tipo mais simples de agente é o **agente reativo simples**. Esses agentes selecionam ações com base na percepção *atual*, ignorando o restante do histórico de percepções. Por exemplo, o agente aspirador de pó cuja função de agente é tabulada na Figura 2.3 é um agente reativo simples, porque sua decisão se baseia apenas na posição atual e no fato de essa posição conter ou não sujeira. Um programa de agente para esse agente é mostrado na Figura 2.8.

Note que o programa de agente aspirador de pó na realidade é muito pequeno em comparação com a tabela correspondente. A redução mais óbvia vem de se ignorar o histórico de percepções, o que reduz o número de possibilidades de 4^T para apenas 4. Uma pequena redução adicional vem do fato de que, quando o quadrado atual está sujo, a ação não depende da posição em que o agente esteja.

Imagine-se como o motorista do táxi automatizado. Se o carro da frente frear e suas luzes de freio se acenderem, você deve notar esse fato e começar a frear. Em outras palavras, algum processamento é realizado de acordo com a entrada visual para estabelecer a condição que chamamos de "O carro da frente está freando". Então, isso ativa alguma conexão estabelecida no programa do agente para a ação "começar a frear". Chamaremos tal conexão **regra condição-ação**,[7] escrita como:

REGRA CONDIÇÃO-AÇÃO

se *carro-da-frente-está-freando* **então** *começar-a-frear*

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

se *estado*= *Sujo* **então retorna** *Aspirar*
senão se *posição* = *A* **então retorna** *Direita*
senão se *posição* = *B* **então retorna** *Esquerda*

Figura 2.8 O programa de agente para um agente reativo simples no ambiente de aspirador de pó de dois estados. Esse programa implementa a função de agente tabulada na Figura 2.3.

**Nota do revisor técnico:* Como aqui se trata de "pseudocódigo" é possível traduzir os comandos para facilitar a leitura. Apenas não se traduz quando se trata de uma linguagem de programação real.

7. Também chamadas **regras situação-ação**, **regras de produção** ou **regras se-então**.

Os seres humanos também têm muitas dessas conexões, algumas das quais são respostas aprendidas (como dirigir) e outras são reflexos inatos (como piscar quando algo se aproxima de seu olho). No decorrer do livro, veremos várias maneiras diferentes de aprender e implementar tais conexões.

O programa da Figura 2.8 é específico para um determinado ambiente de aspirador de pó. Uma abordagem mais geral e flexível consiste em primeiro construir um interpretador de uso geral para regras condição-ação e depois criar conjuntos de regras para ambientes de tarefas específicos. A Figura 2.9 fornece a estrutura desse programa geral em forma esquemática, mostrando como as regras condição-ação permitem ao agente fazer a conexão entre percepção e ação. (Não se preocupe com o fato de esse assunto parecer trivial; ele ficará mais interessante em breve.) Utilizamos retângulos para denotar o estado interno atual do processo de decisão do agente e elipses para representar as informações suplementares usadas no processo. O programa de agente, que também é muito simples, é mostrado na Figura 2.10. A função INTERPRETAR-ENTRADA gera uma descrição abstrata do estado atual da percepção, e a função REGRA-CORRESPONDENTE retorna a primeira regra no conjunto de regras que corresponde à descrição de estado dada. Observe que a descrição em termos de "regras" e "correspondência" é puramente conceitual; as implementações reais podem ser tão simples quanto uma coleção de portas lógicas que implementam um circuito booleano.

Os agentes reativos simples têm a admirável propriedade de serem simples, mas se caracterizam por ter inteligência muito limitada. O agente da Figura 2.10 funcionará *somente se a decisão correta puder ser tomada com base apenas na percepção atual – ou seja, apenas se o ambiente for completamente observável.* Até mesmo uma pequena impossibilidade de observação pode causar sérias dificuldades. Por exemplo, a regra de frenagem apresentada anteriormente pressupõe que a condição *carro-da-frente-está-freando* pode ser determinada a partir do percepto atual – a imagem de vídeo atual – se o carro da frente tiver uma luz de freio central. Infelizmente, modelos mais antigos têm configu-

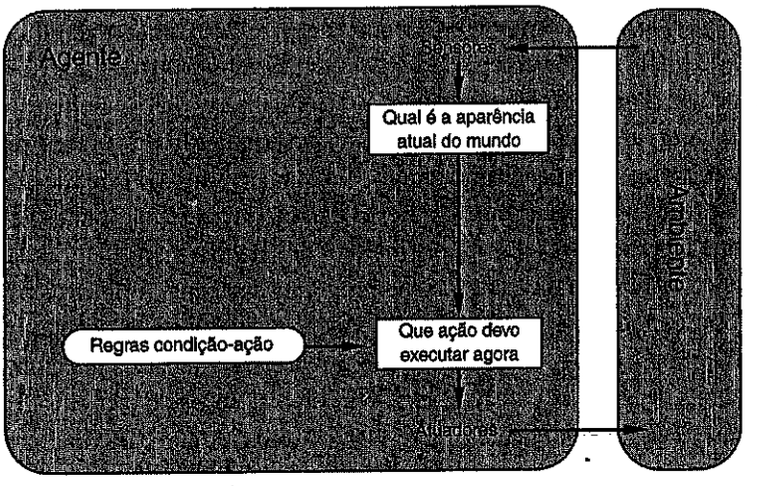

Figura 2.9 Diagrama esquemático de um agente reativo simples.

```
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
```

Figura 2.10 Um agente reativo simples. Ele age de acordo com uma regra cuja condição corresponde ao estado atual definido pela percepção.

rações diferentes de lanternas, luzes de freio e luzes de setas, e nem sempre é possível saber por uma única imagem se o carro está freando. Um agente reativo simples que dirigisse atrás de um carro desse tipo frearia contínua e desnecessariamente ou, pior ainda, nunca frearia.

Podemos ver um problema semelhante surgindo no mundo de aspirador de pó. Suponha que um agente aspirador de pó reativo simples seja destituído de seu sensor de posição, e tenha apenas um sensor de sujeira. Tal agente tem apenas duas percepções possíveis: [*Sujo*] e [*Limpo*]. Ele pode *Aspirar* em resposta a [*Sujo*]; o que deve fazer em resposta a [*Limpo*]? Mover-se para a *Esquerda* falhará (sempre) se ele começar no quadrado *A*, e mover-se para a *Direita* falhará (sempre) se ele começar no quadrado *B*. Com freqüência, laços de repetição infinitos são inevitáveis no caso de agentes reativos simples operando em ambientes parcialmente observáveis.

ALEATORIEDADE

É possível escapar de laços de repetição infinitos se o agente puder tornar suas ações **aleatórias**. Por exemplo, se o agente aspirador de pó perceber [*Limpo*], ele pode jogar uma moeda para escolher entre *Esquerda* e *Direita*. É fácil mostrar que o agente alcançará o outro quadrado usando duas etapas em média. Em seguida, se esse quadrado estiver sujo, ele limpará a sujeira e a tarefa de limpeza será concluída. Conseqüentemente, um agente reativo simples aleatório poderia superar um agente reativo simples determinístico.

Mencionamos na Seção 2.3 que um comportamento aleatório do tipo correto pode ser racional em alguns ambientes multiagentes. Em ambientes de um único agente, em geral a aleatoriedade *não* é racional. Ela é um artifício útil que ajuda um agente reativo simples em algumas situações mas, na maioria dos casos, podemos fazer muito melhor com agentes determinísticos mais sofisticados.

Agentes reativos baseados em modelos

ESTADO INTERNO

O modo mais efetivo de lidar com a possibilidade de observação parcial é o agente *controlar a parte do mundo que ele não pode ver agora*. Isto é, o agente deve manter algum tipo de **estado interno** que dependa do histórico de percepções e assim reflita pelo menos alguns dos aspectos não-observados do estado atual. Para o problema do freio, o estado interno não é muito extenso – apenas o quadro anterior da câmera, que permite ao agente detectar quando duas luzes vermelhas na borda do veículo acendem ou apagam ao mesmo tempo. No caso de outras tarefas de direção, como trocar de pista, o agente precisa controlar onde os outros carros estão, se não puder vê-los todos de uma vez.

A atualização dessas informações internas de estado à medida que o tempo passa exige que dois tipos de conhecimento sejam codificados no programa de agente. Primeiro, precisamos de algumas informações sobre o modo como o mundo evolui independentemente do agente – por exemplo, que um carro que estiver ultrapassando em geral estará mais próximo do que estava um momento antes. Em segundo lugar, precisamos de algumas informações sobre como as ações do próprio agente afetam o mundo – de que, por exemplo, quando o agente girar o volante à direita, o carro irá virar para a direita ou de que, depois de dirigir por cinco minutos na direção norte da auto-estrada, em geral ficamos cinco quilômetros ao norte de onde nos encontrávamos cinco minutos antes. Esse conhecimento de "como o mundo funciona" – seja ele implementado em circuitos booleanos simples ou em teorias científicas completas – é chamado de **modelo** do mundo. Um agente que usa tal modelo denomina-se **agente baseado em modelo**.

AGENTE BASEADO EM MODELO

A Figura 2.11 fornece a estrutura do agente reativo com seu estado interno, mostrando como a percepção atual é combinada com o estado interno antigo para gerar a descrição atualizada do estado atual. O programa de agente é mostrado na Figura 2.12. A parte interessante é a função ATUALIZAR-ESTADO, responsável pela criação da descrição do novo estado interno. Da mesma forma como ele interpreta a nova percepção à luz do conhecimento existente sobre o estado, ele também utiliza informações a respeito da evolução do mundo para controlar as partes não-vistas do mundo, e também deve saber como as ações do agente atuam sobre o estado do mundo. Exemplos detalhados aparecem nos Capítulos 10 e 17.

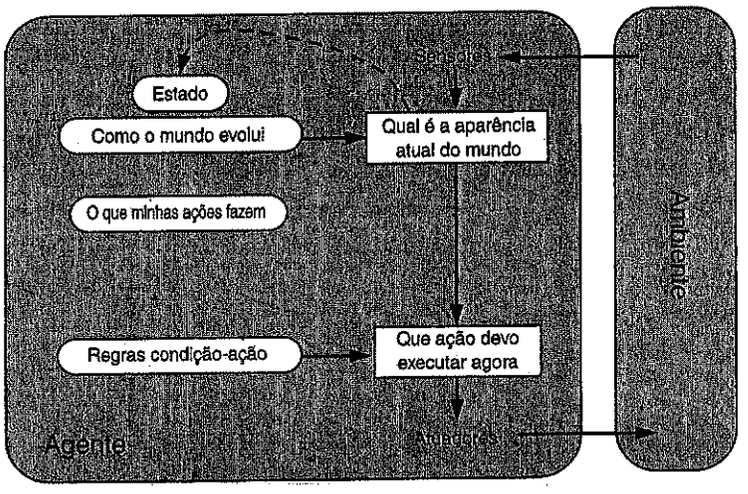

Figura 2.11 Um agente reativo baseado em modelo.

```
função AGENTE-REATIVO-COM-ESTADOS(percepção) retorna uma ação
  variáveis estáticas: estado, uma descrição do estado atual do mundo
        regras, um conjunto de regras condição-ação
        ação, a ação mais recente, inicialmente nenhuma

  estado ← ATUALIZAR-ESTADO(estado, ação, percepção)
  regra ← REGRA-CORRESPONDENTE (estado, regras)
  ação ← AÇÃO-DA-REGRA[regra]
  retornar ação
```

Figura 2.12 Um agente reativo baseado em modelo. Ele controla o estado atual do mundo usando um modelo interno. Em seguida, ele escolhe uma ação da mesma maneira que o agente reativo simples.

Agentes baseados em objetivos

Conhecer o estado atual do ambiente nem sempre é suficiente para se decidir o que fazer. Por exemplo, em um entroncamento de estradas, o táxi pode virar à esquerda, virar à direita ou seguir em frente. A decisão correta depende de onde o táxi está tentando chegar. Em outras palavras, da mesma forma que o agente precisa de uma descrição do estado atual, ele também precisa de alguma espécie de informação sobre **objetivos** que descreva situações desejáveis – por exemplo, estar no destino do passageiro. O programa de agente pode combinar isso com informações sobre os resultados de ações possíveis (as mesmas informações que foram usadas para atualizar o estado interno no agente reativo), a fim de escolher ações que alcancem o objetivo. A Figura 2.13 mostra a estrutura do agente baseado em objetivos.

OBJETIVO

Às vezes, a seleção da ação baseada em objetivos é direta, quando a satisfação do objetivo resulta de imediato de uma única ação. Outras vezes ela será mais complicada, quando o agente tiver de considerar longas seqüências de ações até encontrar um meio de atingir o objetivo. **Busca** (Capítulos 3 a 6) e **planejamento** (Capítulos 11 e 12) são os subcampos da IA dedicados a encontrar seqüências de ações que alcançam os objetivos do agente.

Note que a tomada de decisões desse tipo é fundamentalmente distinta das regras condição-ação descritas antes, pelo fato de envolver consideração do futuro – tanto de "O que acontecerá se eu fizer isso e aquilo?" e "Isso me fará feliz?". Nos projetos de agentes reativos, essas informações não são representadas de forma explícita, porque as regras internas fazem o mapeamento direto de percepções para ações. O agente reativo freia quando vê luzes de freio. Em princípio, um agente baseado em

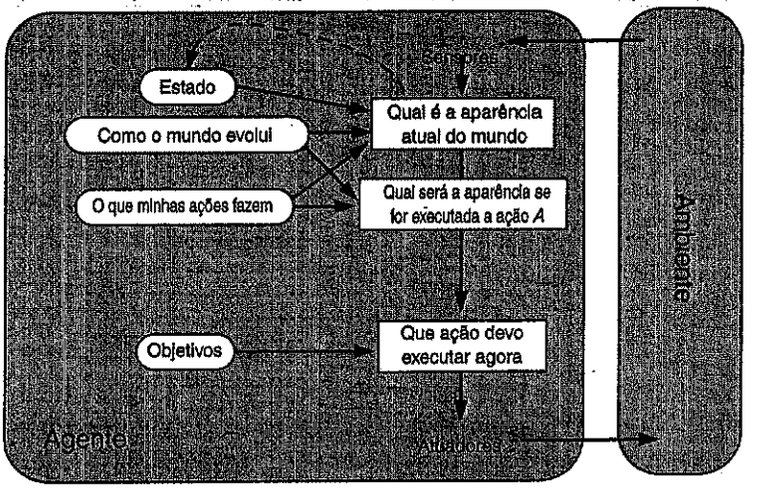

Figura 2.13 Um agente baseado em modelos e orientado para objetivos. Ele controla o estado do mundo, bem como um conjunto de objetivos que está tentando atingir e escolhe uma ação que (no final) levará à realização de seus objetivos.

objetivos poderia raciocinar que, se o carro da frente tem suas luzes de freio acesas, ele diminuirá a velocidade. Dada a forma como o mundo costuma evoluir, a única ação que alcançará o objetivo de não atingir outros carros é frear.

Embora o agente baseado em objetivos pareça menos eficiente, ele é mais flexível, porque o conhecimento que apóia suas decisões é representado de maneira explícita e pode ser modificado. Se começar a chover, o agente poderá atualizar seu conhecimento de como seus freios irão operar de modo eficiente; isso fará todos os comportamentos relevantes serem alterados automaticamente para atender às novas condições. Por outro lado, para o agente reativo, teríamos de reescrever muitas regras condição-ação. O comportamento do agente baseado em objetivos pode ser alterado com facilidade de modo a ir até uma posição diferente. As regras do agente reativo sobre quando fazer curvas e quando seguir em frente só funcionarão para um único destino; todas elas terão de ser substituídas se for preciso ir para algum outro lugar.

Agentes baseados na utilidade

Sozinhos, os objetivos não são realmente suficientes para gerar um comportamento de alta qualidade na maioria dos ambientes. Por exemplo, existem muitas seqüências de ações que levarão o táxi até seu destino (alcançando assim o objetivo), mas algumas são mais rápidas, mais seguras, mais confiáveis ou mais econômicas que outras. Os objetivos simplesmente permitem uma distinção binária crua entre "estados felizes" e "infelizes", enquanto uma medida de desempenho mais geral deve permitir uma comparação entre diferentes estados do mundo, de acordo com o grau exato de felicidade que proporcionariam ao agente se pudessem ser alcançados. Tendo em vista que "feliz" não soa muito científico, a terminologia habitual é dizer que, se um estado do mundo for preferido em detrimento de outro, ele terá maior **utilidade** para o agente.

UTILIDADE

FUNÇÃO DE UTILIDADE

Uma **função de utilidade** mapeia um estado (ou uma seqüência de estados) em um número real, que descreve o grau de felicidade associado. Uma especificação completa da função de utilidade permite decisões racionais em dois tipos de casos nos quais os objetivos são inadequados. Primeiro, quando existem objetivos contraditórios, dos quais apenas alguns podem ser atingidos (por exemplo, velocidade e segurança), a função de utilidade especifica o compromisso apropriado. Em segundo lugar, quando existem vários objetivos que o agente deseja alcançar e nenhum deles pode ser atingido com certeza, a utilidade fornece um meio pelo qual a probabilidade de sucesso pode ser ponderada em relação à importância dos objetivos.

No Capítulo 16, mostraremos que qualquer agente racional deve se comportar *como se* possuísse uma função de utilidade cujo valor esperado ele tenta maximizar. Portanto, um agente que possui uma função de utilidade *explícita* pode tomar decisões racionais, e pode fazê-lo por meio de um algoritmo de uso geral que não depende da função de utilidade específica que está sendo maximizada. Desse modo, a definição "global" de racionalidade – designando-se como racionais as funções de agentes que têm o desempenho mais alto – é transformada em uma restrição "local" sobre projetos de agentes racionais que podem ser expressos em um programa simples.

A estrutura de agente baseado na utilidade aparece na Figura 2.14. Os programas de agentes baseados na utilidade são examinados na Parte V, em que projetamos agentes de tomada de decisões que devem lidar com a incerteza inerente aos ambientes parcialmente observáveis.

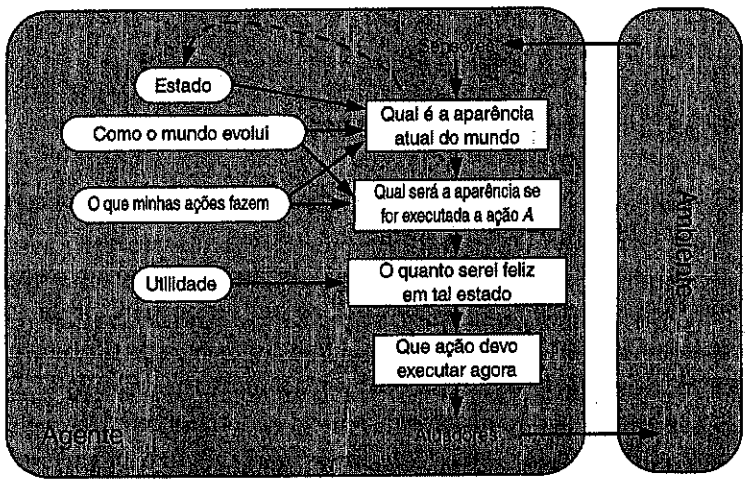

Figura 2.14 Um agente baseado em modelo e orientado para a utilidade. Ele usa um modelo do mundo juntamente com uma função de utilidade que mede suas preferências entre estados do mundo. Em seguida, ele escolhe a ação que leva à melhor utilidade esperada, na qual a utilidade esperada é calculada pela média entre todos os estados resultantes possíveis, ponderados pela probabilidade do resultado.

Agentes com aprendizagem

Descrevemos programas de agentes com vários métodos para selecionar ações. Porém, até agora não explicamos como os programas de agentes *passam a existir*. Em seu famoso ensaio inicial, Turing (1950) considera a idéia de realmente programar suas máquinas inteligentes à mão. Ele estima quanto trabalho isso poderia exigir e conclui que "algum método mais eficiente parece desejável". O método que ele propõe é construir máquinas com aprendizagem e depois ensiná-las. Em muitas áreas de IA, esse é agora o método preferencial para se criar sistemas do estado da arte. O aprendizado tem outra vantagem, como observamos antes: ele permite ao agente operar em ambientes inicialmente desconhecidos e se tornar mais competente do que seu conhecimento inicial sozinho poderia permitir. Nesta seção, introduzimos rapidamente as principais idéias de agentes com aprendizagem. Em quase todos os capítulos do livro, faremos comentários sobre oportunidades e métodos de aprendizado em tipos específicos de agentes. A Parte VI estuda com muito maior profundidade os diversos algoritmos de aprendizado propriamente ditos.

ELEMENTO DE APRENDIZADO
ELEMENTO DE DESEMPENHO
CRÍTICO

Um agente de aprendizado pode ser dividido em quatro componentes conceituais, como mostra a Figura 2.15. A distinção mais importante se dá entre o **elemento de aprendizado**, responsável pela execução de aperfeiçoamentos, e o **elemento de desempenho**, responsável pela seleção de ações externas. O elemento de desempenho é o que antes consideramos como sendo o agente completo: ele recebe percepções e decide sobre ações. O elemento de aprendizado utiliza realimentação do **crítico**

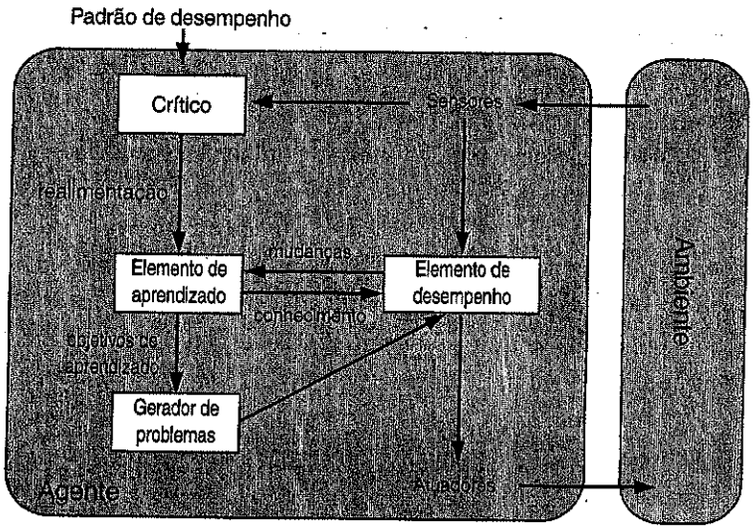

Figura 2.15 Um modelo geral de agentes com aprendizagem.

sobre como o agente está funcionando e determina de que maneira o elemento de desempenho deve ser modificado para funcionar melhor no futuro.

O projeto do elemento de aprendizado depende muito do projeto do elemento de desempenho. Quando se tenta projetar um agente que aprende uma certa capacidade, a primeira pergunta não é "Como farei com que ele aprenda isso?" mas sim "Que tipo de elemento de desempenho meu agente precisará ter para fazer isso depois de ter aprendido como fazê-lo?". Dado um projeto de agente, podem ser construídos mecanismos de aprendizado para otimizar cada parte do agente.

O crítico informa ao elemento de aprendizado como o agente está se comportando em relação a um padrão fixo de desempenho. O crítico é necessário, porque as próprias percepções não oferecem nenhuma indicação do sucesso do agente. Por exemplo, um programa de xadrez poderia receber uma percepção indicando que ele aplicou um xeque-mate em seu oponente, mas o programa precisa de um padrão de desempenho para saber que isso é algo bom; a percepção em si não diz nada sobre isso. É importante que o padrão de desempenho seja fixo. Conceitualmente, deveríamos pensar nele como algo que está totalmente fora do agente, porque o agente não deve modificá-lo para ajustá-lo a seu próprio comportamento.

GERADOR DE PROBLEMAS

O último componente do agente com aprendizagem é o **gerador de problemas**. Ele é responsável por sugerir ações que levarão a experiências novas e informativas. A questão é que, se o elemento de desempenho tivesse a possibilidade, ele continuaria a realizar as melhores ações, dadas as informações que possui. Porém, se o agente estivesse disposto a realizar uma pequena exploração e executar algumas ações que talvez não fossem ótimas a curto prazo, ele poderia descobrir ações muito melhores a longo prazo. A tarefa do gerador de problemas é sugerir essas ações exploratórias. É isso que os cientistas fazem quando realizam experiências. Galileu não pensava que soltar pedras do alto de uma torre em Pisa teria algum valor em si. Ele não estava tentando quebrar as pedras, nem modificar os cérebros de pedestres desafortunados. Seu objetivo era modificar seu próprio cérebro, identificando uma teoria melhor sobre o movimento dos objetos.

Para tornar o projeto global mais concreto, vamos voltar ao exemplo do táxi automatizado. O elemento de desempenho consiste em qualquer coleção de conhecimento e procedimentos que o táxi tem para selecionar suas ações de dirigir. O táxi vai para a estrada e dirige, usando esse elemento de desempenho. O crítico observa o mundo e repassa informações ao elemento de aprendizado. Por exemplo, depois que o táxi faz uma rápida mudança para a esquerda cruzando três faixas de tráfego, o crítico observa a linguagem chocante utilizada por outros motoristas. A partir dessa experiência, o elemento de aprendizado é capaz de formular uma regra afirmando que essa foi uma ação ruim, e o

elemento de desempenho é modificado pela instalação da nova regra. O gerador de problemas pode identificar certas áreas de comportamento que necessitam de melhorias e sugerir experimentos, como testar os freios em diferentes superfícies de rodagem sob condições distintas.

O elemento de aprendizado pode fazer mudanças em qualquer dos componentes de "conhecimento" mostrados nos diagramas de agentes (Figuras 2.9, 2.11, 2.13 e 2.14). Os casos mais simples envolvem o aprendizado direto a partir da seqüência de percepções. A observação de pares de estados sucessivos do ambiente pode permitir ao agente aprender "Como o mundo evolui", e a observação dos resultados de suas ações pode permitir que ele aprenda "O que minhas ações fazem". Por exemplo, se o táxi exercer uma certa pressão nos freios ao dirigir em uma estrada molhada, ele logo descobrirá qual é a desaceleração realmente alcançada. É claro que essas duas tarefas de aprendizado serão mais difíceis se o ambiente for apenas parcialmente observável.

As formas de aprendizado no parágrafo anterior não precisam ter acesso ao padrão de desempenho externo – de certo modo, o padrão universal é fazer previsões que concordem com a experiência. A situação é um pouco mais complexa no caso de um agente baseado em utilidade que deseja aprender informações de utilidade. Por exemplo, suponha que o agente de direção de táxi não receba dos passageiros nenhuma dica de que o táxi sacolejou muito durante a viagem. O padrão de desempenho externo deve informar ao agente que a falta de dicas é uma contribuição negativa para seu desempenho global; desse modo, o agente talvez fosse capaz de aprender que manobras violentas não contribuem para sua própria utilidade. De certo modo, o padrão de desempenho distingue parte da percepção de entrada como uma **recompensa** (ou **penalidade**) que fornece realimentação direta sobre a qualidade do comportamento do agente. Os padrões de desempenho internos como dor e fome em animais podem ser entendidos desse modo. Essa questão é discutida com maior profundidade no Capítulo 21.

Em resumo, os agentes têm uma variedade de componentes, e esses componentes podem ser representados de muitas formas dentro do programa do agente; dessa forma, parece haver grande variedade de métodos de aprendizado. No entanto, existe um único tema unificador. O aprendizado em agentes inteligentes pode ser resumido como um processo de modificação de cada componente do agente, a fim de levar os componentes a um acordo mais íntimo com as informações de realimentação disponíveis, melhorando assim o desempenho global do agente.

2.5 Resumo

Este capítulo foi uma espécie de excursão vertiginosa pela IA, que concebemos como a ciência de projeto de agentes. Aqui estão os pontos importantes a serem lembrados:

- Um **agente** é algo que percebe e age em um ambiente. A **função de agente** relativa a um agente especifica a ação executada pelo agente em resposta a qualquer seqüência de percepções.
- A **medida de desempenho** avalia o comportamento do agente em um ambiente. Um **agente racional** age para maximizar o valor esperado da medida de desempenho, dada a seqüência de percepções vista até o momento.
- Uma especificação de **ambiente de tarefa** inclui a medida de desempenho, o ambiente externo, os atuadores e os sensores. Ao se projetar um agente, o primeiro passo sempre deve ser especificar o ambiente de tarefa de maneira tão completa quanto possível.
- Os ambientes de tarefas variam ao longo de diversas dimensões significativas. Eles podem ser completa ou parcialmente observáveis, determinísticos ou estocásticos, episódicos ou seqüenciais, estáticos ou dinâmicos, discretos ou contínuos e de agente único ou multiagentes.

- O **programa de agente** implementa a função de agente. Existe uma variedade de projetos básicos de programas de agentes, refletindo a espécie de informações explicitadas e usadas no processo de decisão. Os projetos variam em eficiência, densidade e flexibilidade. O projeto apropriado do programa de agente depende da natureza do ambiente.
- Os **agentes reativos simples** respondem diretamente a percepções, enquanto os **agentes reativos baseados em modelos** mantêm o estado interno para controlar aspectos do mundo que não estão evidentes na percepção atual. Os **agentes baseados em objetivos** agem para alcançar seus objetivos, e os **agentes baseados em utilidade** tentam maximizar sua própria "felicidade" esperada.
- Todos os agentes podem melhorar seu desempenho por meio do **aprendizado**.

Notas bibliográficas e históricas

O papel central da ação na inteligência – a noção de raciocínio prático – remonta pelo menos à época da *Ética a Nicômaco* de Aristóteles. O raciocínio prático também foi o assunto do importante artigo de McCarthy (1958), "Programs with Common Sense". Os campos da robótica e da teoria de controle, por sua própria natureza, se preocupam principalmente com a elaboração de agentes físicos. O conceito de **controlador** em teoria de controle é idêntico ao de um agente em IA. Talvez seja surpreendente o fato de a IA ter se concentrado, durante a maior parte de sua história, em componentes de agentes isolados – sistemas de resposta a perguntas, demonstradores de teoremas, sistemas de visão e assim por diante – em lugar de agentes completos. A discussão de agentes no texto de Genesereth e Nilsson (1987) foi uma exceção importante. A visão do agente como um todo é agora extensamente aceita no campo e é um tema central em textos recentes (Poole *et al.*, 1998; Nilsson, 1998).

CONTROLADOR

O Capítulo 1 identificou as raízes do conceito de racionalidade na filosofia e na economia. Em IA, o conceito era de interesse periférico até meados da década de 1980, quando começou a suscitar muitas discussões sobre os fundamentos técnicos próprios do campo. Um ensaio de Jon Doyle (1983) previu que o projeto de agentes racionais seria visto como a missão central da IA, enquanto outros tópicos populares acabariam por constituir novas disciplinas.

A atenção cuidadosa às propriedades do ambiente e suas conseqüências para o projeto de agentes racionais aparecem melhor na tradição da teoria de controle – por exemplo, sistemas de controle clássicos (Dorf e Bishop, 1999) lidam com ambientes completamente observáveis e determinísticos; o controle ótimo estocástico (Kumar e Varaiya, 1986) trata de ambientes estocásticos parcialmente observáveis, e o controle híbrido (Henzinger e Sastry, 1998) lida com ambientes que contêm elementos discretos e elementos contínuos. A distinção entre ambientes completa e parcialmente observáveis também é central na literatura de **programação dinâmica** desenvolvida na área de pesquisa operacional (Puterman, 1994), que discutiremos no Capítulo 17.

Os agentes reativos constituíram o modelo fundamental para behavioristas psicológicos como Skinner (1953), que tentou reduzir a psicologia de organismos estritamente a mapeamentos de entrada/saída ou estímulo/resposta. O avanço desde o behaviorismo até o funcionalismo em psicologia, que foi pelo menos em parte orientado pela aplicação da metáfora de computadores a agentes (Putnam, 1960; Lewis, 1966), inseriu no quadro o estado interno do agente. A maioria dos trabalhos relacionados à IA visualiza a idéia de agentes reativos puros com estado como algo demasiado simples para proporcionar grande avanço, mas o trabalho de Rosenschein (1985) e Brooks (1986) questionou essa suposição (consulte o Capítulo 25). Nos últimos anos, grande parte do trabalho se dedicou à busca de algoritmos eficientes para controlar ambientes complexos (Hamscher *et al.*, 1992). O programa Remote Agent que controlava a astronave Deep Space One (descrita na página 28) é um exemplo particularmente impressionante (Muscettola *et al.*, 1998; Jonsson *et al.*, 2000).

Os agentes baseados em objetivos são pressupostos em tudo desde a visão de Aristóteles do raciocínio prático até os primeiros ensaios de McCarthy sobre a IA baseada em lógica. Shakey the Robot (Fikes e Nilsson, 1971; Nilsson, 1984) foi a primeira encarnação robótica de um agente lógico baseado em objetivos. Uma análise lógica completa sobre agentes baseados em objetivos foi apresentada em Genesereth e Nilsson (1987), e uma metodologia de programação baseada em objetivos, denominada programação orientada a agentes, foi desenvolvida por Shoham (1993).

A visão baseada em objetivos também domina a tradição da psicologia cognitiva na área de resolução de problemas, começando com o influente trabalho *Human Problem Solving* (Newell e Simon, 1972) e passando por todo o trabalho mais recente de Newell (Newell, 1990). Os objetivos, analisados com maior profundidade como *desejos* (gerais) e *intenções* (buscadas no momento), são centrais para a teoria de agentes desenvolvida por Bratman (1987). Essa teoria tem sido influente tanto para a compreensão da linguagem natural quanto para sistemas multiagentes.

Horvitz *et al.* (1988) sugerem especificamente o uso da racionalidade concebida como a maximização da utilidade esperada, como base para a IA. O texto de Pearl (1988) foi o primeiro em IA a abordar em profundidade a teoria de probabilidade e utilidade; sua exposição de métodos práticos para raciocínio e tomada de decisões sob incerteza talvez tenha sido o maior fator para o rápido deslocamento em direção a agentes baseados em utilidade nos anos 90 (veja a Parte V).

O projeto geral de agentes com aprendizagem representado na Figura 2.15 é clássico na literatura de aprendizagem de máquinas (Buchanan *et al.*, 1978; Mitchell, 1997). Exemplos do projeto, materializados em programas, remontam no mínimo ao programa de aprendizado de Arthur Samuel (1959, 1967) para jogar damas. Os agentes com aprendizagem são descritos em profundidade na Parte VI.

O interesse em agentes e em projeto de agentes cresceu com rapidez nos últimos anos, em parte devido ao crescimento da Internet e à necessidade percebida de **softbots** automatizados e móveis (Etzioni e Weld, 1994). Documentos relevantes estão reunidos em *Readings in Agents* (Huhns e Singh, 1998), e em *Foundations of Rational Agency* (Wooldridge e Rao, 1999). *Multiagent Systems* (Weiss, 1999) fornece uma base sólida para muitos aspectos do projeto de agentes. As conferências dedicadas a agentes incluem a International Conference on Autonomous Agents, o International Workshop on Agent Theories, Architectures, and Languages e a International Conference on Multiagent Systems. Finalmente, *Dung Beetle Ecology* (Hanski e Cambefort, 1991) fornece uma grande quantidade de informações interessantes sobre o comportamento de besouros de esterco.

Exercícios

2.1 Defina com suas próprias palavras os termos a seguir: agente, função de agente, programa de agente, racionalidade, autonomia, agente reativo, agente baseado em modelo, agente baseado em objetivos, agente baseado em utilidade, agente com aprendizagem.

2.2 Tanto a medida de desempenho quanto a função de utilidade medem o quanto um agente está desempenhando bem suas atividades. Explique a diferença entre as duas medidas.

2.3 Este exercício explora as diferenças entre funções de agentes e programas de agentes.

a. Pode haver mais de um programa de agente que implemente uma dada função de agente? Dê um exemplo ou mostre por que não é possível.

b. Existem funções de agentes que não podem ser implementadas por qualquer programa de agente?

c. Dada uma arquitetura de máquina fixa, cada programa de agente implementa exatamente uma função de agente?

d. Dada uma arquitetura com n bits de armazenamento, quantos programas de agentes distintos são possíveis?

2.4 Vamos examinar a racionalidade de várias funções do agente aspirador de pó.

a. Mostre que a função do agente aspirador de pó simples descrito na Figura 2.3 é realmente racional, conforme as suposições listadas na página 37.

b. Descreva uma função de agente racional para a medida de desempenho modificada que deduz um ponto a cada movimento. O programa de agente correspondente exige estado interno?

c. Descreva possíveis projetos de agentes para os casos em que quadrados limpos podem ficar sujos e a geografia do ambiente é desconhecida. Faz sentido para o agente aprender a partir de sua experiência nessas situações? Em caso afirmativo, o que ele deve aprender?

2.5 Para cada um dos agentes a seguir, desenvolva uma descrição de PEAS do ambiente de tarefa:

a. Robô jogador de futebol.

b. Agente catálogo de compras da Internet.

c. Andarilho autônomo de Marte.

d. Assistente de matemático para demonstração de teoremas.

2.6 Para cada um dos tipos de agentes listados no Exercício 2.5, caracterize o ambiente de acordo com as propriedades dadas na Seção 2.3 e selecione um projeto de agente adequado.

Todos os exercícios a seguir estão relacionados à implementação de ambientes e agentes para o mundo de aspirador de pó.

2.7 Implemente um simulador de ambiente de medição de desempenho para o mundo de aspirador de pó representado na Figura 2.2 e especificado na página 37. Sua implementação deve ser modular, de forma que os sensores, os atuadores e as características do ambiente (tamanho, forma, localização da sujeira etc.) possam ser alterados com facilidade. (*Nota:* Para algumas opções de linguagens de programação e sistemas operacionais, já existem implementações no repositório de código on-line.)

2.8 Implemente um agente reativo simples para o ambiente de aspirador de pó no Exercício 2.7. Execute o simulador de ambiente com esse agente para todas as configurações iniciais possíveis de sujeira e posições do agente. Registre a pontuação de desempenho do agente para cada configuração e sua pontuação média global.

2.9 Considere uma versão modificada do ambiente de aspirador de pó do Exercício 2.7, na qual o agente é penalizado com um ponto para cada movimento.

a. Um agente reativo simples pode ser perfeitamente racional para esse ambiente? Explique.

b. E um agente reativo com estado? Projete tal agente.

c. Como suas respostas para os itens **a** e **b** mudarão se as percepções do agente fornecerem o *status* limpo/sujo de cada quadrado no ambiente?

2.10 Considere uma versão modificada do ambiente de aspirador de pó do Exercício 2.7, na qual a geografia do ambiente – extensão, limites e obstáculos – é desconhecida, como também a configuração inicial de sujeira. (O agente também pode se mover *Acima* e *Abaixo*, além de *Esquerda* e *Direita*.)

a. Um agente reativo simples pode ser perfeitamente racional para esse ambiente? Explique.

b. Um agente reativo simples com uma função de agente *aleatório* pode superar um agente reativo simples? Projete tal agente e faça a medição de seu desempenho em vários ambientes.

c. Você poderia projetar um ambiente no qual seu agente aleatório terá um desempenho muito ruim? Mostre seus resultados.

d. Um agente reativo com estado pode superar um agente reativo simples? Projete tal agente e faça a medição de seu desempenho em vários ambientes. Você pode projetar um agente racional desse tipo?

2.11 Repita o Exercício 2.10 para o caso em que o sensor de posição é substituído por um sensor de "impacto" que descobre as tentativas realizadas pelo agente para se mover em um obstáculo ou cruzar os limites do ambiente. Suponha que o sensor de impacto pare de funcionar; como o agente deverá se comportar?

2.12 Os ambientes de aspiradores de pó de todos os exercícios anteriores eram determinísticos. Descreva possíveis programas de agentes para cada uma das versões estocásticas listadas a seguir:

a. Lei de Murphy: durante 25% do tempo, a ação de *Aspirar* deixa de limpar o chão se ele está sujo e deposita sujeira no chão se ele está limpo. De que maneira seu programa de agente é afetado se o sensor de sujeira fornece a resposta errada durante 10% do tempo?

b. Crianças pequenas: em cada período de tempo, cada quadrado limpo tem uma chance de 10% de ficar sujo. Você poderia apresentar um projeto de agente racional para esse caso?