هوش مصنوعی

فصل ۲

عامل‌های هوشمند

Intelligent Agents

کاظم فولادی قلعه
دانشکده مهندسی، دانشکدگان فارابی
دانشگاه تهران

http://courses.fouladi.ir/ai

بسم الله الرحمن الرحیم

هوش مصنوعی

درس ۷

عامل‌های هوشمند (۱)

Intelligent Agents (1)

کاظم فولادی قلعه
دانشکده مهندسی، دانشکدگان فارابی
دانشگاه تهران

http://courses.fouladi.ir/ai

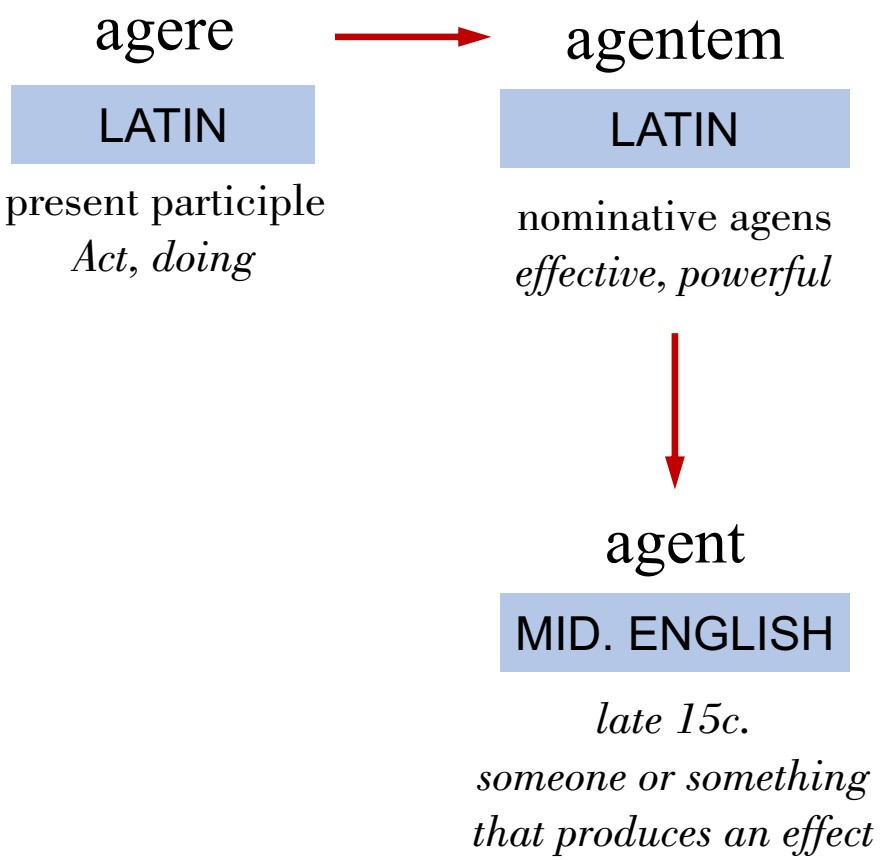
هوش مصنوعی

عامل‌های هوشمند

◆

مقدمه

اتیمولوژی: Agent



http://www.etymonline.com/index.php?term=agent

ترمینولوژی: Agent

Full Definition of *Agent*

1 : one that acts or exerts power

2 :

a : something that produces or is capable of producing an effect :
an active or efficient cause Education proved to be an agent of change in the community.

b : a chemically, physically, or biologically active principle an oxidizing agent

3 : a means or instrument by which a guiding intelligence achieves a result

4 : one who is authorized to act for or in the place of another: such as

a : a representative, emissary, or official of a government crown agent federal agent

b : one engaged in undercover activities (such as espionage) : spy a secret agent

c : a business representative (as of an athlete or entertainer) a theatrical agent

5 : a computer application designed to automate certain tasks (such as gathering information online)

http://www.merriam-webster.com/dictionary/agent

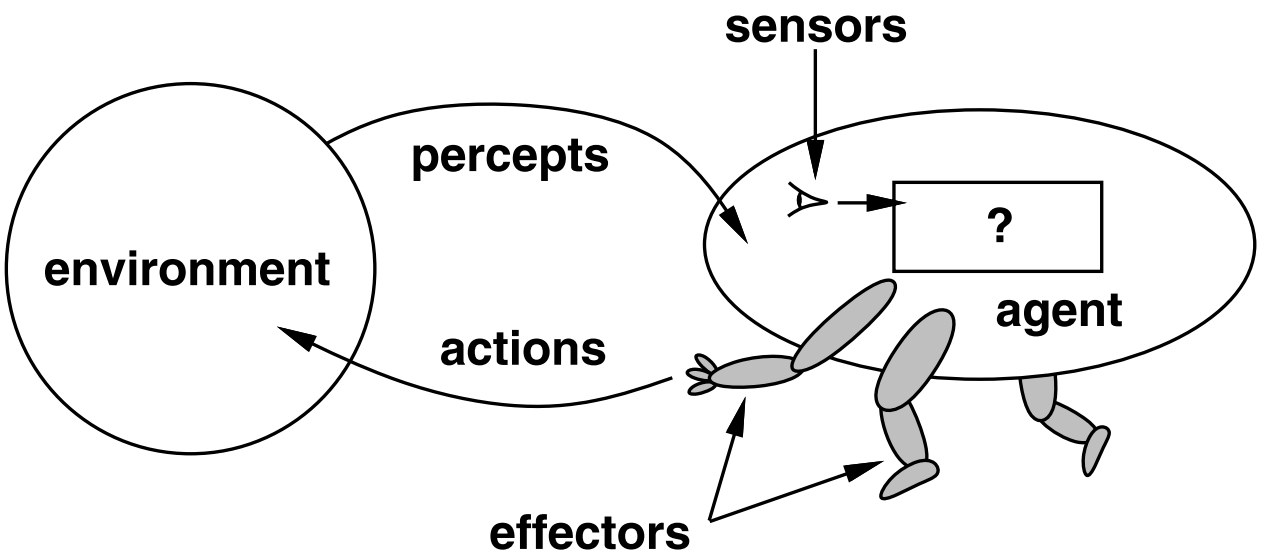
هوش مصنوعی

عامل‌های هوشمند

۱

عامل‌ها
و
محیط‌ها

عامل

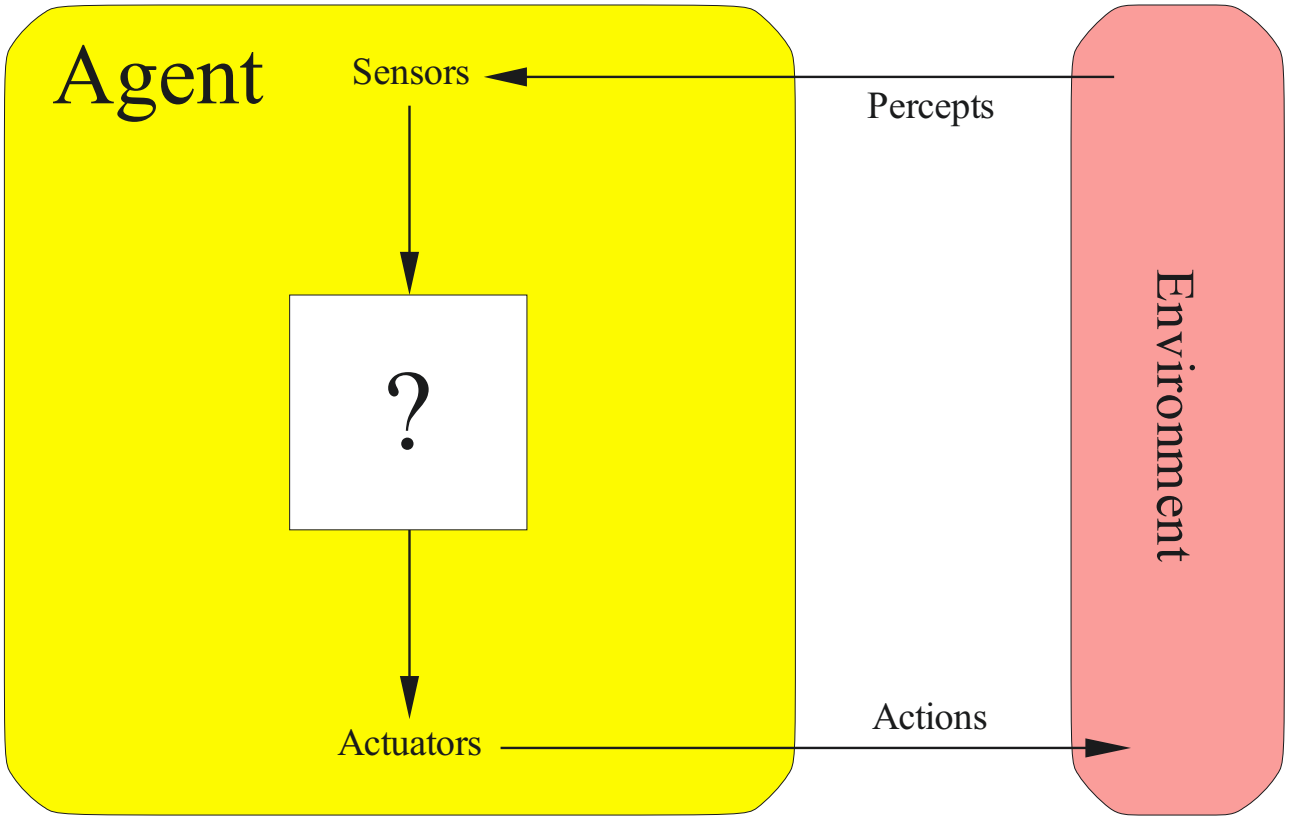


عامل موجودیتی است که در محیط کار انجام می‌دهد.

عامل

AGENT

عامل چیزی است که می‌تواند «به این صورت دیده شود» که
محیط را از طریق حسگرهای خود درک می‌کند و
سپس از طریق کنش‌گرهای خود روی آن کنش انجام می‌دهد.



عامل

مثال

عامل انسانی *Human agent*	حسگر	حواس پنج‌گانه: چشم، گوش و ...
	کنش‌گر	دست و پا، زبان و ...
عامل روباتی *Robotic agent*	حسگر	دوربین، فاصله‌یاب مادون قرمز و ...
	کنش‌گر	موتور و ...
عامل نرم‌افزاری *Software agent*	حسگر	دنباله‌ای از بیت‌ها
	کنش‌گر	دنباله‌ای از بیت‌های کد شده

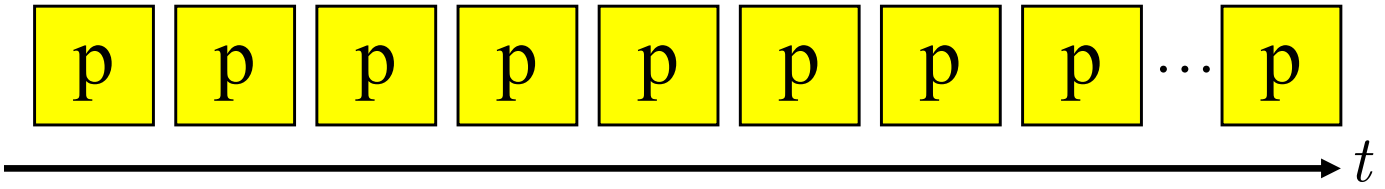
مفهوم **عامل** وسیله‌ای برای مطالعه‌ی هوش مصنوعی است.

تقسیم‌بندی اشیا به «عامل» و «غیرعامل» نادرست است.
بر اساس نگاه طراح هر چیزی می‌تواند یک «عامل» باشد.

ادراک و دنباله‌ی ادراکی

ادراک (percept)، ورودی‌های ادراکی عامل در هر لحظه‌ی داده شده است.

دنباله‌ی ادراکی (percept sequence)، تاریخچه‌ی کامل همه‌ی چیزهایی است که عامل درک کرده است.



انتخاب کنش در هر لحظه توسط عامل،
می‌تواند به کل دنباله‌ی ادراکی مشاهده شده تا آن لحظه وابسته باشد،
اما به هیچ چیزی که تا آن لحظه آن را درک نکرده است، وابسته نیست.

تابع عامل

AGENT FUNCTION

رفتار عامل، با **تابع عامل** توصیف می‌شود که
هر **دنباله‌ی ادراکی** داده شده را به یک **کنش** نگاشت می‌دهد.

$$f : \mathcal{P}^* \longrightarrow \mathcal{A}$$

تابع عامل

مجموعه‌ی همه‌ی
دنباله‌های ادراکی

مجموعه‌ی همه‌ی
کنش‌های عامل

برنامه‌ی عامل

AGENT PROGRAM

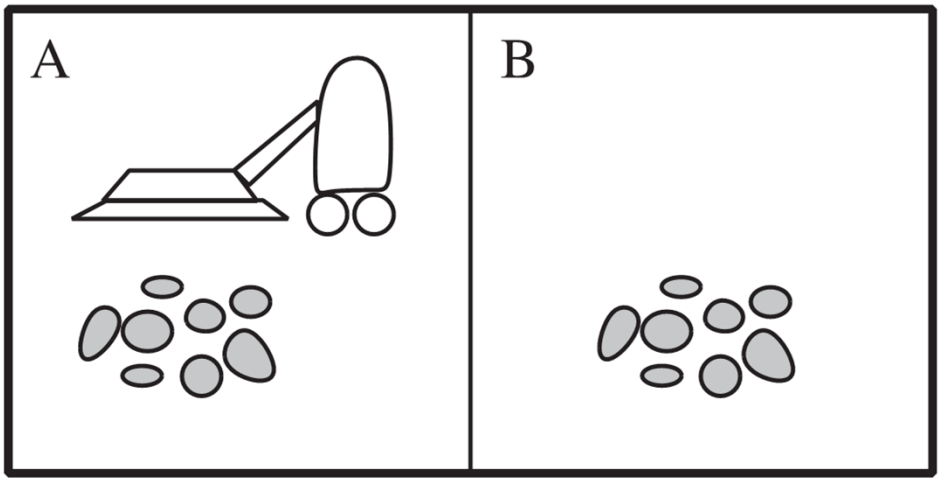
برنامه‌ی عامل، تابع عامل را بر روی یک معماری فیزیکی پیاده‌سازی و اجرا می‌کند.

نسبت تابع عامل با برنامه‌ی عامل

برنامه‌ی عامل *Agent Program*	تابع عامل *Agent Function*
انضمامی ـ مجسم (concrete)	انتزاعی ـ مجرد (abstract)
برنامه‌نویسی	ریاضی
ورودی: ادراک لحظه‌ای	ورودی: یک دنباله‌ی ادراکی
قابل اجرا روی یک معماری	-
یک تابع عامل، می‌تواند چند برنامه‌ی عامل داشته باشد	ممکن است یک تابع عامل، برنامه‌ی عامل نداشته باشد

مثال: دنیای جاروبرقی

VACUUM-CLEANER WORLD

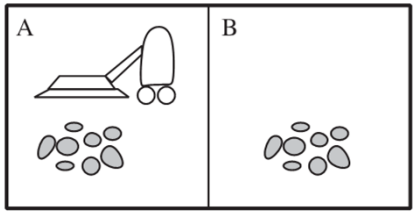


عامل جاروبرقی	
ادراک‌ها *Percepts*	**کنش‌ها** *Actions*
مکان عامل، محتوای آن مکان e.g. $[A, Dirty]$	حرکت‌های ممکن عامل $Left, Right, Suck, NoOp$

یک عامل جاروبرقی

تابع عامل: در قالب جدول

A VACUUM-CLEANER AGENT



Percept sequence	Action
$[A, Clean]$	$Right$
$[A, Dirty]$	$Suck$
$[B, Clean]$	$Left$
$[B, Dirty]$	$Suck$
$[A, Clean], [A, Clean]$	$Right$
$[A, Clean], [A, Dirty]$	$Suck$
⋮	⋮
$[A, Clean], [A, Clean], [A, Clean]$	$Right$
$[A, Clean], [A, Clean], [A, Dirty]$	$Suck$
⋮	⋮

Figure 2.3 Partial tabulation of a simple agent function for the vacuum-cleaner world shown in Figure 2.2.

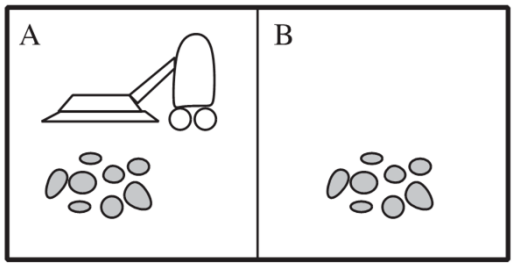
مشکل: این جدول را چگونه باید پر کرد؟

مثال: دنیای جاروبرقی

برنامه‌ی عامل (در قالب کد)

function REFLEX-VACUUM-AGENT([*location*,*status*]) **returns** an action

if *status* = *Dirty* **then return** *Suck*
else if *location* = *A* **then return** *Right*
else if *location* = *B* **then return** *Left*

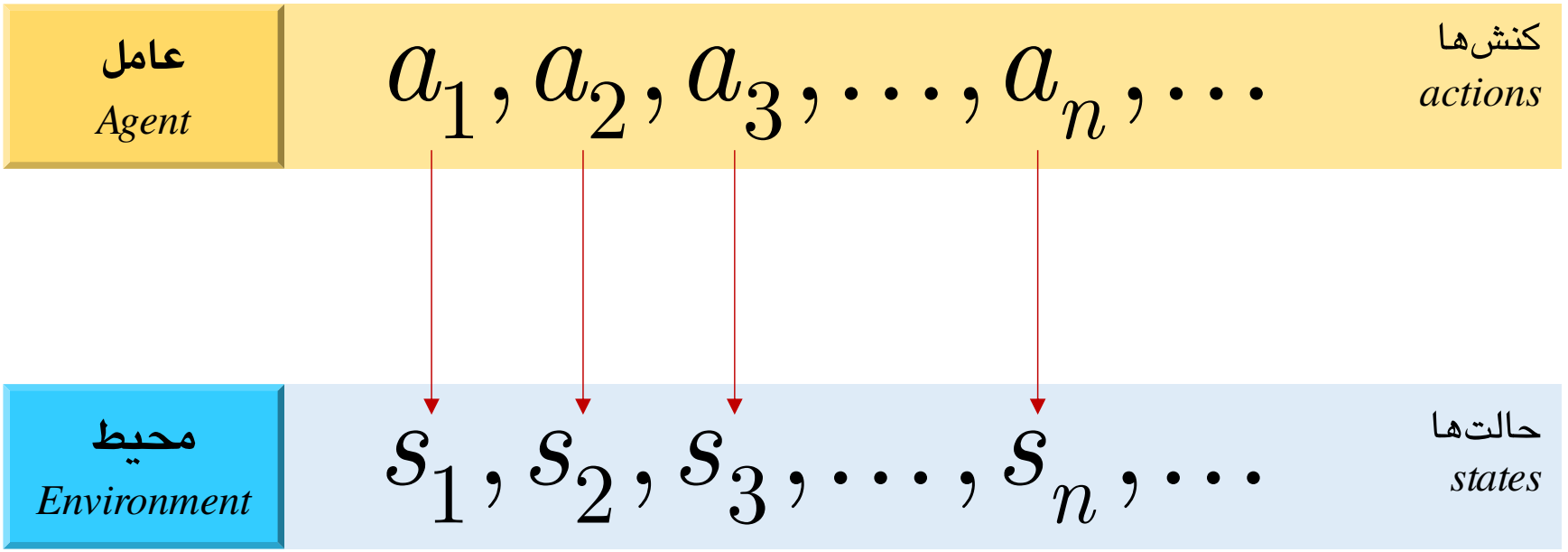


Percept sequence	Action
[*A*, *Clean*]	*Right*
[*A*, *Dirty*]	*Suck*
[*B*, *Clean*]	*Left*
[*B*, *Dirty*]	*Suck*
[*A*, *Clean*], [*A*, *Clean*]	*Right*
[*A*, *Clean*], [*A*, *Dirty*]	*Suck*
⋮	⋮
[*A*, *Clean*], [*A*, *Clean*], [*A*, *Clean*]	*Right*
[*A*, *Clean*], [*A*, *Clean*], [*A*, *Dirty*]	*Suck*
⋮	⋮

Figure 2.3 Partial tabulation of a simple agent function for the vacuum-cleaner world shown in Figure 2.2.

تأثیر عامل بر محیط

کنش‌های عامل ـ حالت‌های محیط



هوش مصنوعی

عامل‌های هوشمند

۲

رفتار خوب: مفهوم رسیونالیته

عامل رسیونال

RATIONAL AGENT

عامل رسیونال:

عاملی که رفتار خوب انجام می‌دهد.

تقریب اول: رفتار خوب؛
رفتاری که باعث بیشترین موفقیت عامل شود.

نیاز به روشی برای اندازه‌گیری موفقیت
(معیار کارآیی)

معیار کارآیی

معیار خوب بودن

PERFORMANCE MEASURE

معیار کارآیی

میزان موفقیت رفتار یک عامل را نشان می‌دهد.

«معیار ارزیابی با در نظر گرفتن پی‌آمدهای رفتار آن عامل»

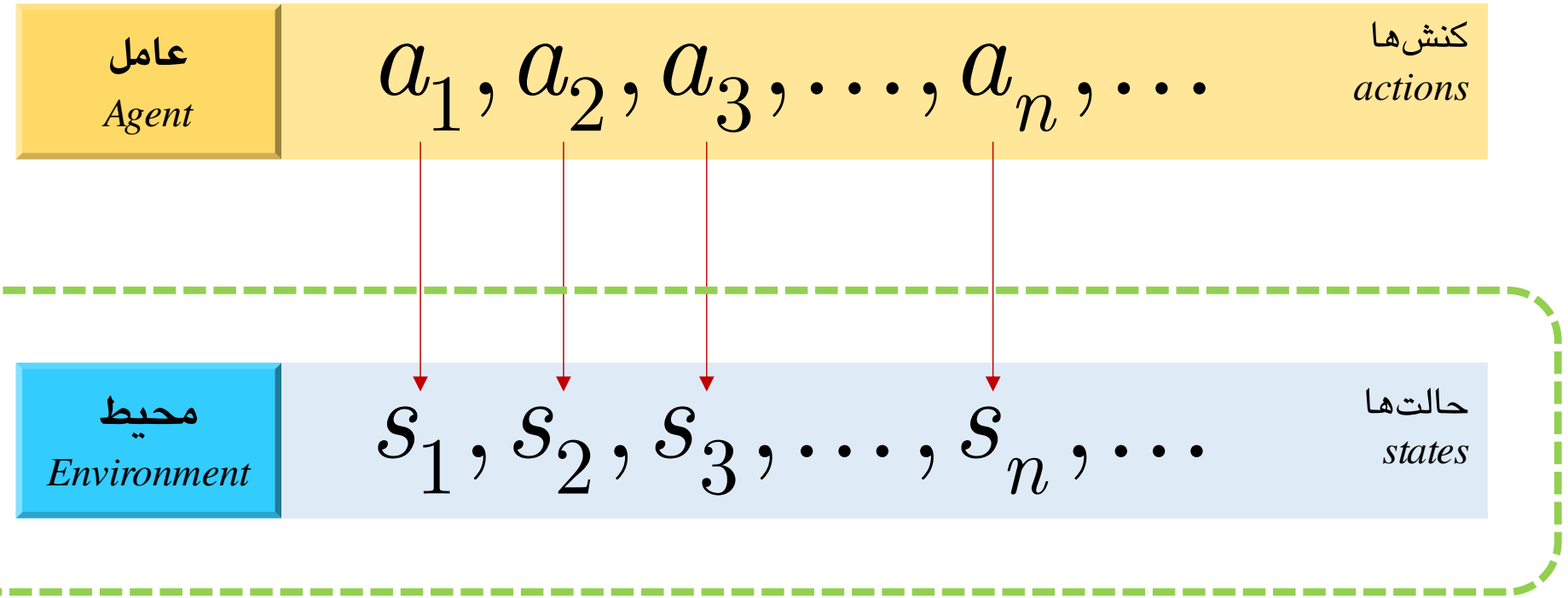
هر عامل در یک محیط بر اساس ادراک‌هایی که دریافت می‌کند، یک دنباله از کنش‌ها تولید می‌کند؛
دنباله‌ی کنش‌های عامل، موجب دنباله‌ای از حالت‌های محیط می‌شود:
اگر این دنباله‌ی حالت‌های محیط، مطلوب بود، آن‌گاه عامل به خوبی عمل کرده است.

مفهوم مطلوب بودن در معیار کارآیی احصا می‌شود:
معیار کارآیی هر دنباله از حالت‌های محیط را ارزیابی می‌کند.

تذکر: در معیار کارآیی، حالت‌های محیط مهم است، نه حالت‌های عامل
یعنی معیار کارآیی را بر اساس آنچه در محیط می‌خواهیم طراحی می‌کنیم،
نه براساس آنچه فکر می‌کنیم عامل باید انجام بدهد.

معیار کارآیی

بر اساس دنباله‌ی حالت‌های محیط



معیار کارآیی

پی‌آمدگرایی

CONSEQUENTIALISM

پی‌آمدگرایی (نتیجه‌گرایی)

روی‌کرد پذیرفته شده در هوش مصنوعی برای مفهوم «چیز درست»

«ارزیابی رفتار یک عامل از طریق پی‌آمدهای رفتار آن عامل»

فلسفه‌ی اخلاق چندین تصور مختلف از مفهوم «چیز درست» را توسعه داده است،
اما هوش مصنوعی به طور کلی به یک مفهوم به نام «پی‌آمدگرایی» پایبند شده است.

معیار کارآیی

ملاحظات

مقایسه‌ی معیار کارآیی عینی و ذهنی	
معیار کارآیی ذهنی *Subjective Performance Measure*	**معیار کارآیی عینی** *Objective Performance Measure*
نظر شخصی خود عامل برای ارزیابی او	نظر محیط برای ارزیابی عامل (توسط طراح و سازنده‌ی عامل)
متغیر: قابل تغییر توسط خود عامل	ثابت: غیر قابل تغییر توسط عامل
معیار کارآیی با نیت موفق‌تر شدن عامل انتخاب می‌شود	معیار کارآیی بر اساس هدف انتخاب می‌شود
مشکل: ممکن است عامل نتواند پاسخ بدهد	مزیت: نیازی به پرسش از خود عامل نیست
مشکل: ممکن است عامل پاسخ گمراه‌کننده و اشتباه بدهد	مزیت: عامل نمی‌تواند بسادگی محیط را فریب بدهد

مشخصات یک معیار کارآیی مناسب

- **نباید** عامل را به دلیل یک چیز غیر قابل درک جریمه کند.
- **نباید** عامل را به دلیل یک کنش غیر قابل انجام جریمه کند.
- **نباید** عامل را به دلیل یک پی‌آمد غیر قابل پیش‌بینی جریمه کند.
- **باید** ارزیابی عامل در طول مدت کار (+ اول + آخر کار) باشد.
- **باید** هر دوی فرآیند و نتیجه‌ی کار عامل را در نظر بگیرد.

یک معیار کارآیی ثابت برای همه‌ی وظیفه‌ها و همه‌ی عامل‌ها وجود ندارد:

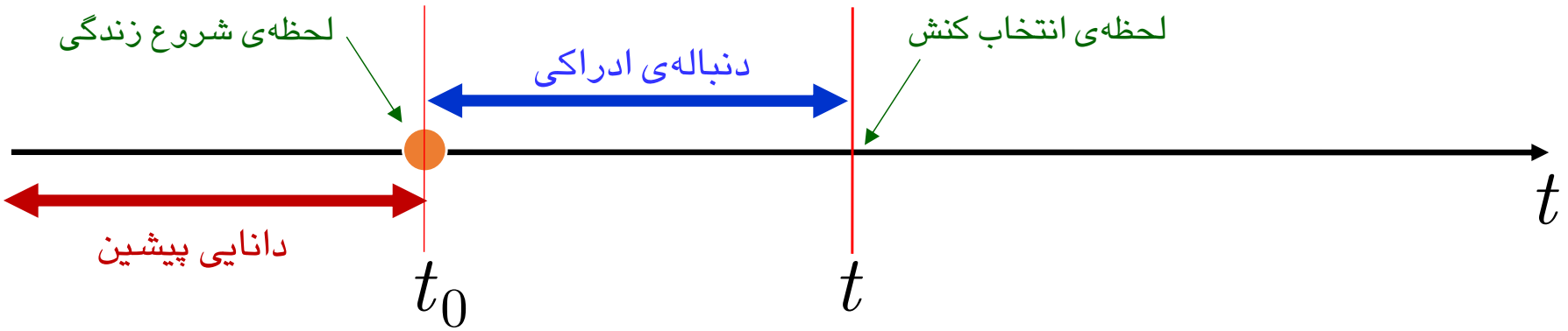
طراح باید متناسب با شرایط، معیار کارآیی را تدبیر کند.

رسیونالیته

پارامترهای وابسته

RATIONALITY

رسیونالیته در هر زمان وابسته به چهار مورد است :			
۱	۲	۳	۴
معیار کارآیی *Performace Measure*	**دانایی پیشینی** *Prior Knowledge*	**کنش‌ها** *Actions*	**دنباله‌ی ادراکی** *Percept Sequence*
ضابطه‌ی موفقیت عامل	دانایی پیشینی عامل از محیط	کنش‌هایی که عامل قادر به انجام آنهاست.	دنباله‌ی ادراکی عامل تا آن زمان



عامل رسیونال

RATIONAL AGENT

یک عامل رسیونال

کنشی را انتخاب می‌کند که
مقدار مورد انتظار معیار کارآیی را
با داشتن دنباله‌ی ادراکی تا آن لحظه
ماکزیمم می‌کند.

عامل رسیونال

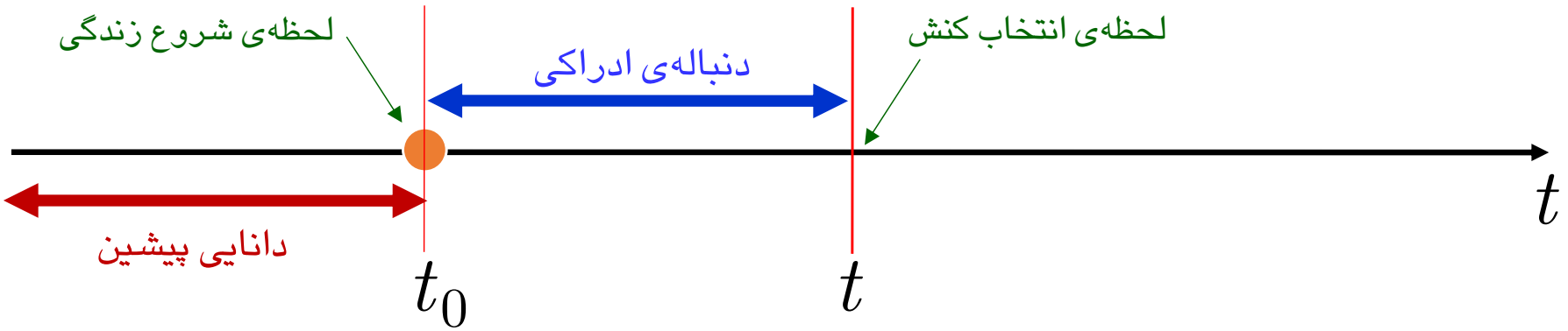
تعریف (راسل و نورویگ)

RATIONAL AGENT

یک عامل رسیونال

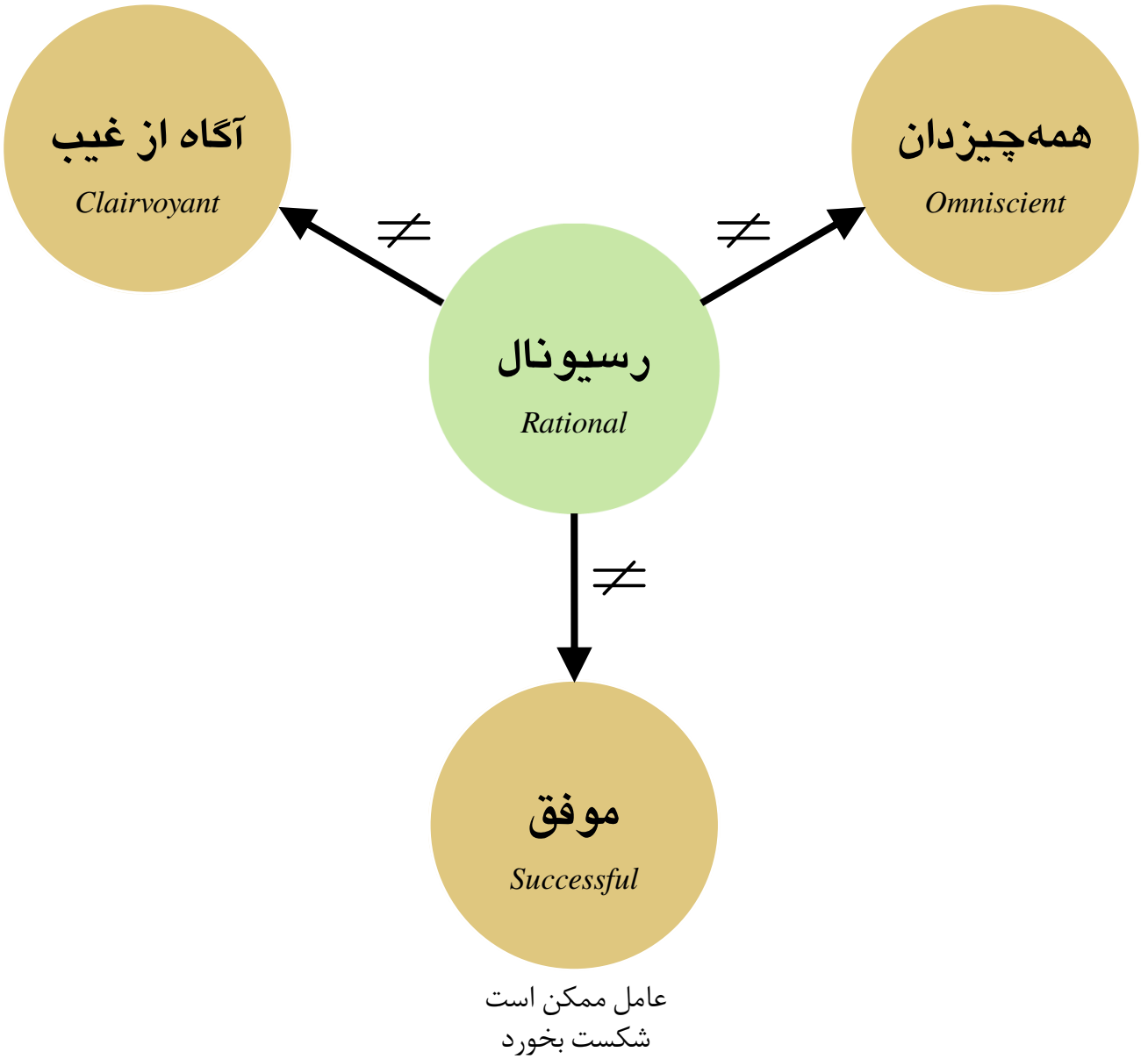
برای هر دنباله‌ی ادراکی ممکن
کنشی را انتخاب می‌کند که
انتظار دارد معیار کارآیی آن را ماکزیمم کند؛

بر اساس شواهدی که توسط دنباله‌ی ادراکی فراهم می‌شود و
آنچه دانایی درون‌سازی شده‌ی عامل است.



رسیونالیته

RATIONALITY



پیامد کنش‌ها ممکن است مورد انتظار نباشد

ادراکات ممکن است همه‌ی اطلاعات مربوط را فراهم نکند

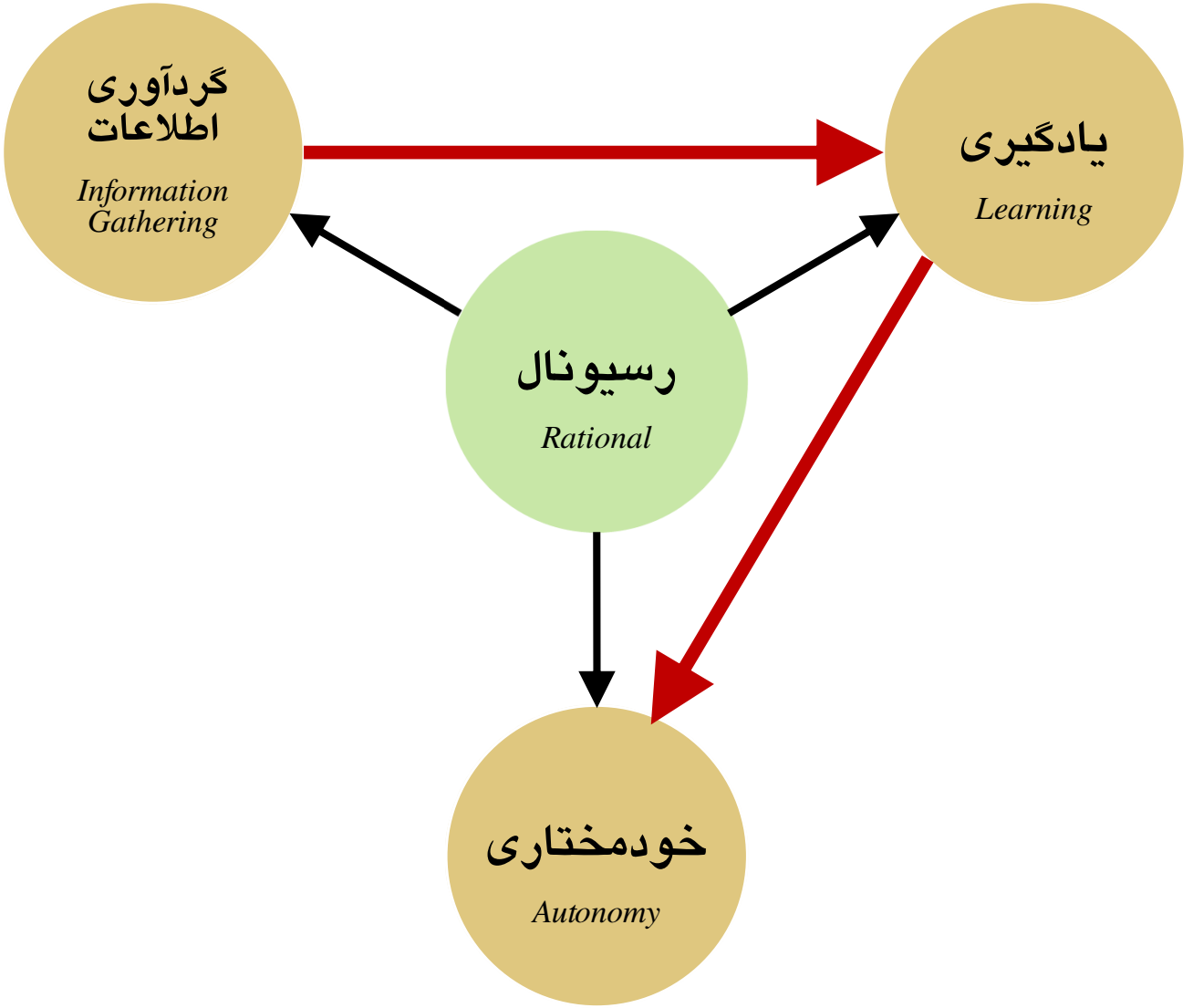
عامل ممکن است شکست بخورد

تمایز رسیونالیته با کامل‌بودن

رسیونالیته *Rationality*	کامل بودن *Perfection*
ماکزیمم‌سازی کارآیی مورد انتظار (expected)	ماکزیمم‌سازی کارآیی واقعی (actual)
انجام بهترین کاری که توانسته است بفهمد	انجام بهترین کار ممکن
پیاده‌سازی امکان‌پذیر	پیاده‌سازی غیر ممکن

رسیونالیته

ملزومات



عامل نباید رفتار غیر هوشمندانه داشته باشد: قبل از تصمیم‌گیری باید اطلاعات کافی از محیط جمع کند

یادگیری از تجربیات و ادراک‌ها

دانایی عامل مستقل از دانایی اولیه‌ی آن می‌شود؛
رفتار عامل توسط تجربه‌ی او تعیین می‌شود.

ملزومات رسیونالیته

(۱) گردآوری اطلاعات

عامل نباید رفتار غیر هوشمندانه داشته باشد:
قبل از تصمیم‌گیری باید اطلاعات کافی از محیط جمع کند

مثل: نگاه کردن به دو طرف خیابان قبل از عبور از آن.

گردآوری اطلاعات: انجام کنش‌هایی به منظور تغییر کنش‌های آینده	
نگاه کردن *Looking*	**اکتشاف** *Exploration*
در محیط شناخته‌شده	در محیط ناشناخته

ملزومات رسیونالیته

(۲) یادگیری

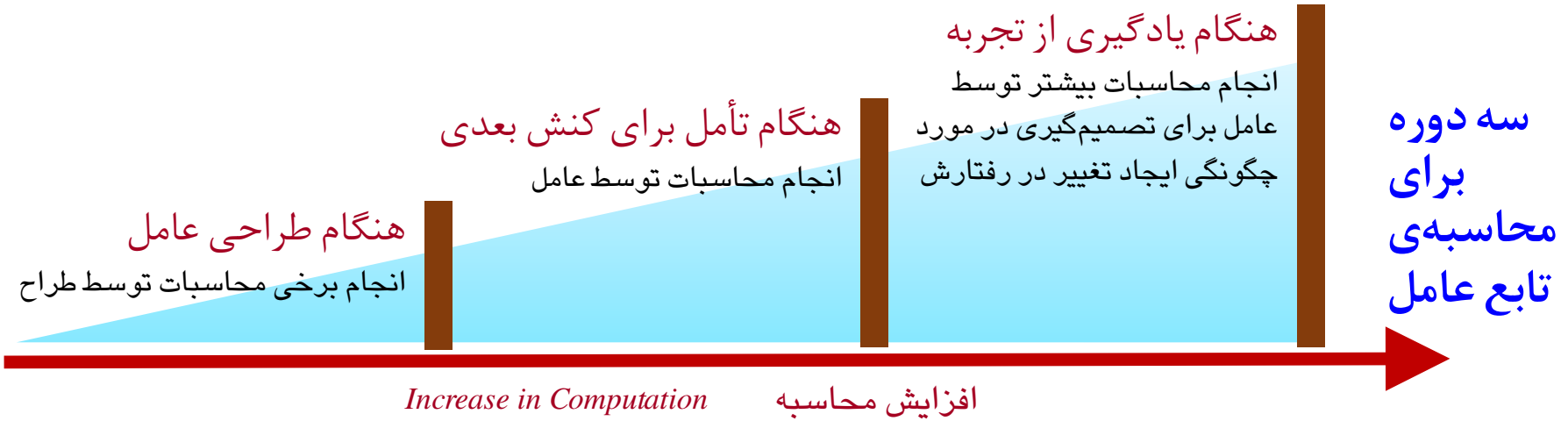
یادگیری

Learning

یادگیری از تجربیات و ادراک‌ها تا حد ممکن

پیکربندی اولیه‌ی عامل: نشان‌دهنده‌ی **دانایی پیشینی** از محیط

پیکربندی ثانویه‌ی عامل: تغییر و اصلاح دانایی عامل بر اساس **تجربه**



ملزومات رسیونالیته

(۳) خودمختاری

خودمختاری

Autonomy

دانایی عامل مستقل از دانایی اولیه‌ی آن می‌شود؛
رفتار عامل توسط تجربه‌ی او تعیین می‌شود.

به میزانی که عامل به جای تجربه‌ی خودش به دانایی پیشینی تعبیه شده در آن توسط طراح تکیه می‌کند، دارای کمبود خودمختاری (آزادی عمل) است.

عامل رسیونال باید خودمختار باشد.
(برای پالایش دانایی غلط یا ناقص پیشینی، باید تا جایی که می‌تواند یاد بگیرد.)

تذکر: در عمل به‌ندرت به خودمختاری کامل از ابتدا نیاز داریم:
وقتی عامل تجربه ندارد، یا تجربه‌ی کمی دارد، باید تصادفی عمل کند، مگر اینکه طراح به او کمک کند.

افزودن یادگیری، طراحی یک عامل رسیونال ساده را ممکن می‌کند که در محیطهای گوناگون موفق خواهد بود.

بسم الله الرحمن الرحیم

هوش مصنوعی

درس ۸

عامل‌های هوشمند (۲)

Intelligent Agents (2)

کاظم فولادی قلعه
دانشکده مهندسی، دانشکدگان فارابی
دانشگاه تهران

http://courses.fouladi.ir/ai

هوش مصنوعی

عامل‌های هوشمند

۳

طبیعت محیط‌ها

مشخص‌سازی محیط وظیفه

SPECIFYING THE TASK ENVIRONMENT

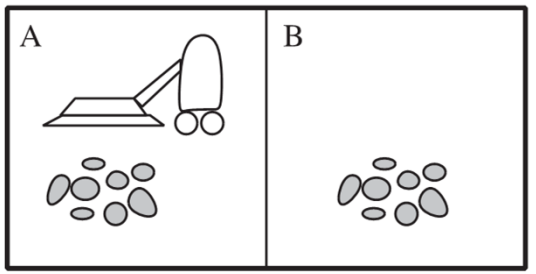
P	E	A	S
معیار کارآیی	محیط	کنش‌گرها	حسگرها
Performace Measure	*Environment*	*Actuators*	*Sensors*

مشخص‌سازی محیط وظیفه

مثال: دنیای جاروبرقی

SPECIFYING THE TASK ENVIRONMENT: THE VACUUM-CLEANER AGENT

P	E	A	S
معیار کارآیی *Performace Measure*	**محیط** *Environment*	**کنش‌گرها** *Actuators*	**حسگرها** *Sensors*
برای هر مربع تمیز در هر گام زمانی، یک امتیاز مثبت کلاً ۱۰۰۰ گام زمانی طول عمر	نقشه‌ی محیط معلوم، توزیع آشغال‌ها و مکان اولیه‌ی عامل نامعلوم	انجام کنش‌های راست چپ مکش هیچ	حسگر مکان حسگر کثیفی مربع



مشخص‌سازی محیط وظیفه

مثال: تاکسی خودکار هوشمند

SPECIFYING THE TASK ENVIRONMENT: AUTOMATIC INTELLIGENT TAXI

P	E	A	S
معیار کارآیی *Performace Measure*	**محیط** *Environment*	**کنش‌گرها** *Actuators*	**حسگرها** *Sensors*
ایمنی رسیدن به مقصد فایده رعایت قانون راحتی ...	خیابان‌ها/آزادراه‌ها ترافیک علایم راهنمایی عابرین پیاده آب و هوا ...	فرمان گاز ترمز بوق بلندگو/نمایشگر ...	تصویر شتاب‌سنج درجه‌ها حسگرهای موتور صفحه‌کلید موقعیت‌سنج GPS ...

مشخص‌سازی محیط وظیفه

مثال : عامل خرید اینترنتی

SPECIFYING THE TASK ENVIRONMENT: INTERNET SHOPPING AGENT

P	E	A	S
معیار کارآیی *Performace Measure*	**محیط** *Environment*	**کنش‌گرها** *Actuators*	**حسگرها** *Sensors*
قیمت کیفیت مناسب بودن کارآمدی ...	سایت‌های وب حال و آینده فروشندگان خریداران ...	نمایش به کاربر دنبال کردن یک URL پر کردن فرم ...	خواننده‌ی صفحات HTML (متن، گرافیک، اسکریپت) ...

خصوصیات محیط‌های وظیفه

PROPERTIES OF TASK ENVIRONMENTS

مشاهده‌پذیر جزئی *Partially Observable*		مشاهده‌پذیر کامل *Fully Observable*
چند عاملی *Multiagent*		تک عاملی *Single-agent*
اتفاقی *Stochastic*	استراتژیک *Strategic*	قطعی *Deterministic*
دنباله‌ای *Sequential*		مقطعی *Episodic*
پویا *Dynamic*	نیمه‌پویا *Semidynamic*	ایستا *Static*
پیوسته *Continuous*		گسسته *Discrete*
ناشناخته *Unknown*		شناخته‌شده *Known*

خصوصیات محیط‌های وظیفه

مشاهده‌پذیر کامل یا مشاهده‌پذیر جزئی

مشاهده‌پذیر جزئی
Partially Observable

حسگرهای عامل دسترسی به جزئی از حالت محیط در هر لحظه را به عامل می‌دهند.

مثلا عدم وجود حسگر، حسگرهای نادقیق، حسگرهای نویزی

مشاهده‌پذیر کامل به طور مؤثر
Effectively Fully Observable

حسگرهای عامل همه‌ی جنبه‌های **مربوط** به انتخاب کنش را نشان می‌دهند.
(مربوط بودن وابسته به معیار کارآیی)

مشاهده‌پذیر کامل
Fully Observable

حسگرهای عامل دسترسی به حالت کامل محیط در هر لحظه را به عامل می‌دهند.

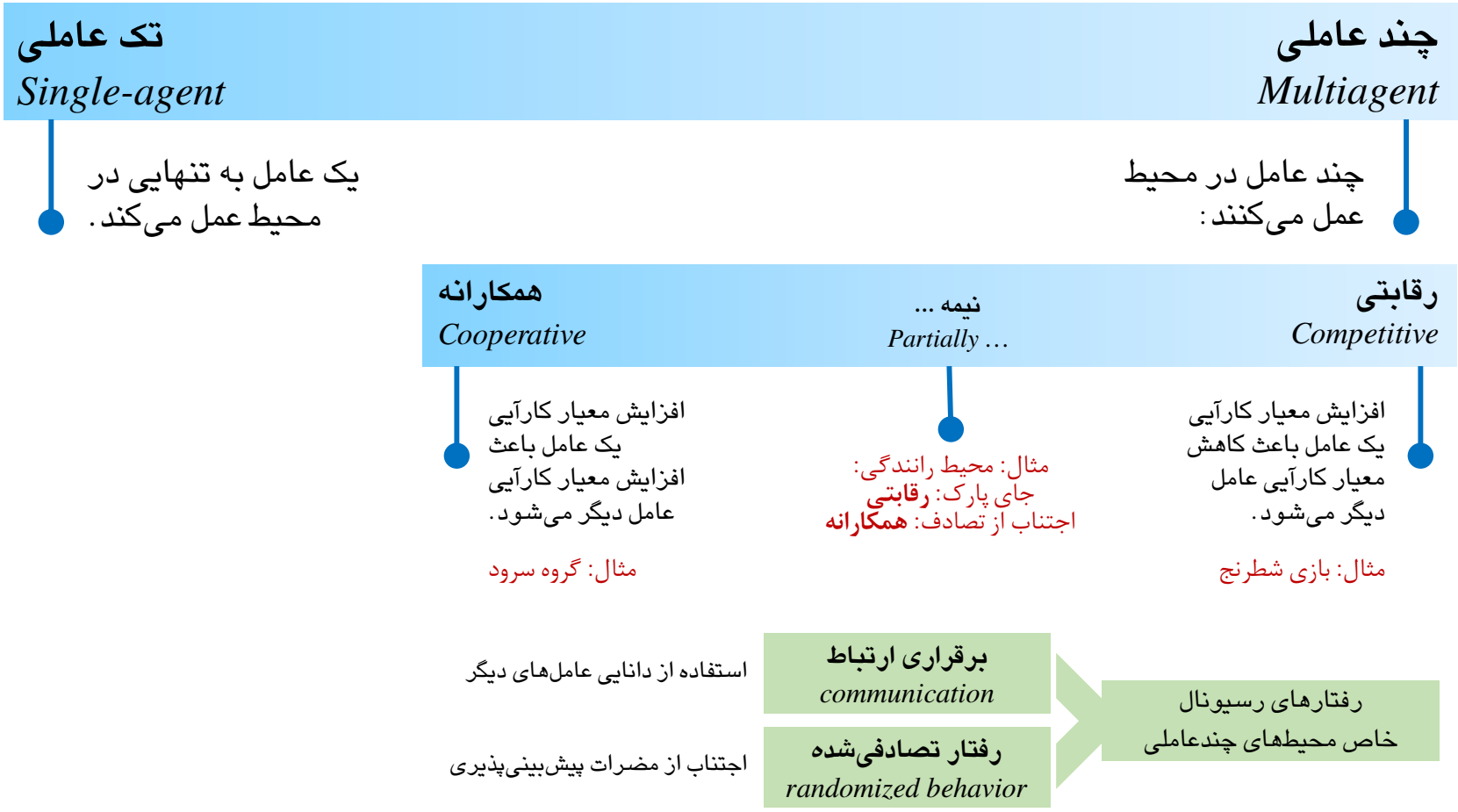
راحتی کار: عامل برای دنبال کردن دنیا نیازی به نگهداری حالت داخلی ندارد.

مشاهده‌ناپذیر
Unobservable

عامل هیچ حسگری ندارد.

خصوصیات محیط‌های وظیفه

تک عاملی یا چند عاملی



کدام موجودیت **می‌تواند** به عنوان عامل دیده شود؟ ⟵ هر چیزی که محیط را درک کند و روی محیط کنش انجام دهد.

کدام موجودیت **می‌باید** به عنوان عامل دیده شود؟ ⟵ هر چیزی که برای ماکزیمم‌سازی معیار کارآیی‌اش، که به رفتار دیگری هم وابسته است، تلاش می‌کند.

خصوصیات محیط‌های وظیفه

قطعی یا اتفاقی

قطعی
Deterministic

حالت بعدی محیط توسط **حالت فعلی** و **کنش** اجرا شده توسط عامل به طور کامل تعیین می‌شود.

راحتی کار: عامل برای دنبال کردن دنیا نیازی به نگهداری حالت داخلی ندارد.

استراتژیک
Strategic

محیط قطعی است، بجز در مورد کنش سایر عامل‌ها

اتفاقی
Stochastic

حالت بعدی محیط توسط **حالت فعلی** و **کنش** اجرا شده توسط عامل به طور کامل تعیین نمی‌شود.

- محیط مشاهده‌پذیر جزئی ممکن است اتفاقی به نظر برسد.
- عدم قطعیت در مورد برآمدها برحسب احتمالات کمی می‌شوند.

نامطمئن
Uncertain

محیطی که مشاهده‌پذیر کامل یا قطعی نباشد.

غیرقطعی
Non-deterministic

حالت بعدی محیط توسط **حالت فعلی** و **کنش** اجرا شده توسط عامل به طور کامل تعیین نمی‌شود.

کنش‌ها با برآمدهای ممکن آنها مشخص می‌شوند، بدون انتساب احتمال به آنها

خصوصیات محیط‌های وظیفه

مقطعی یا دنباله‌ای

دنباله‌ای
Sequential

تصمیم فعلی می‌تواند بر همه‌ی تصمیم‌های آینده تأثیر بگذارد.

کنش‌های کوتاه‌مدت می‌توانند پی‌آمدهای بلندمدت داشته باشند.

مثال: **شطرنج**، **رانندگی تاکسی**

مقطعی
Episodic

تجربه‌ی عامل قابل تقسیم به مقطع‌های اتمیک است.

در هر مقطع، عامل یک ادراک دریافت می‌کند و سپس یک کنش واحد انجام می‌دهد.

مقطع بعدی به کنش‌های انجام شده در مقاطع قبلی وابسته نیست.

انتخاب کنش در هر مقطع فقط وابسته به همان مقطع است.

ساده‌تر است: عامل نیازی ندارد به جلو فکر کند!

مثال: عمده‌ی وظایف **طبقه‌بندی**

خصوصیات محیط‌های وظیفه

ایستا یا پویا

پویا
Dynamic

اگر محیط بتواند در هنگام تأمل عامل تغییر کند.

محیط برای آن عامل، پویا است.

محیط پویا به طور مداوم از عامل می‌پرسد که می‌خواهد چه کنشی را انجام دهد. اگر هنوز تصمیم نگرفته باشد، فرض می‌کند تصمیم گرفته است کاری انجام ندهد.

مثال: رانندگی تاکسی (سایر خودروها و عابرین در حین تصمیم‌گیری عامل، حرکت می‌کنند).

نیمه‌پویا
Semidynamic

اگر خود محیط با گذر زمان تغییر نکند، اما امتیاز کارآیی عامل تغییر کند.

مثال: شطرنج با ساعت

ایستا
Static

اگر محیط نتواند در هنگام تأمل عامل تغییر کند.

محیط برای آن عامل، ایستا است.

ساده‌تر است: (۱) عامل نیازی ندارد در هنگام تصمیم‌گیری در مورد یک کنش به نگاه کردن ادامه دهد. (۲) عامل لازم نیست نگران گذر زمان باشد.

مثال: شطرنج معمولی

خصوصیات محیط‌های وظیفه

گسسته یا پیوسته

پیوسته *Continuous*	**گسسته** *Discrete*
مؤلفه‌های پیوسته در کار هستند.	مؤلفه‌های گسسته در کار هستند.
مثال: رانندگی تاکسی (حالت، زمان، ادراک، کنش)	مثال: شطرنج معمولی (حالت، زمان، ادراک، کنش)

تمایز **پیوسته** و **گسسته**: در مؤلفه‌های
حالت محیط، نحوه‌ی برخورد با زمان، ادراک‌ها و کنش‌های عامل

خصوصیات محیط‌های وظیفه

شناخته‌شده یا ناشناخته

ناشناخته
Unknown

عامل باید یاد بگیرد که چگونه عمل کند تا تصمیم‌های خوبی بگیرد.

یک محیط ناشناخته می‌تواند مشاهده‌پذیر کامل باشد.

مثال: بازی‌های ویدئویی جدید؛
صفحه‌ی بازی می‌تواند کل حالت بازی را نشان بدهد، اما معلوم نیست دکمه‌ها چه کاری انجام می‌دهند که با آزمایش معلوم می‌شود.

تمایز میان محیط شناخته‌شده و ناشناخته، همان تمایز بین محیط مشاهده‌پذیر کامل و جزئی نیست.

شناخته‌شده
Known

برآمدها (یا احتمال برآمدها در محیط اتفاقی) برای همه‌ی کنش‌ها داده شده است.

یک محیط شناخته‌شده می‌تواند مشاهده‌پذیر جزئی باشد.

مثال: بازی کارت

این خصوصیت، فقط به خود محیط برنمی‌گردد، بلکه
به **حالت دانایی عامل** (یا طراح آن) در مورد «**قوانین فیزیک**» **محیط** برمی‌گردد.

خصوصیات محیط‌های وظیفه

مثال‌هایی از محیط‌های وظیفه و مشخصه‌های آنها

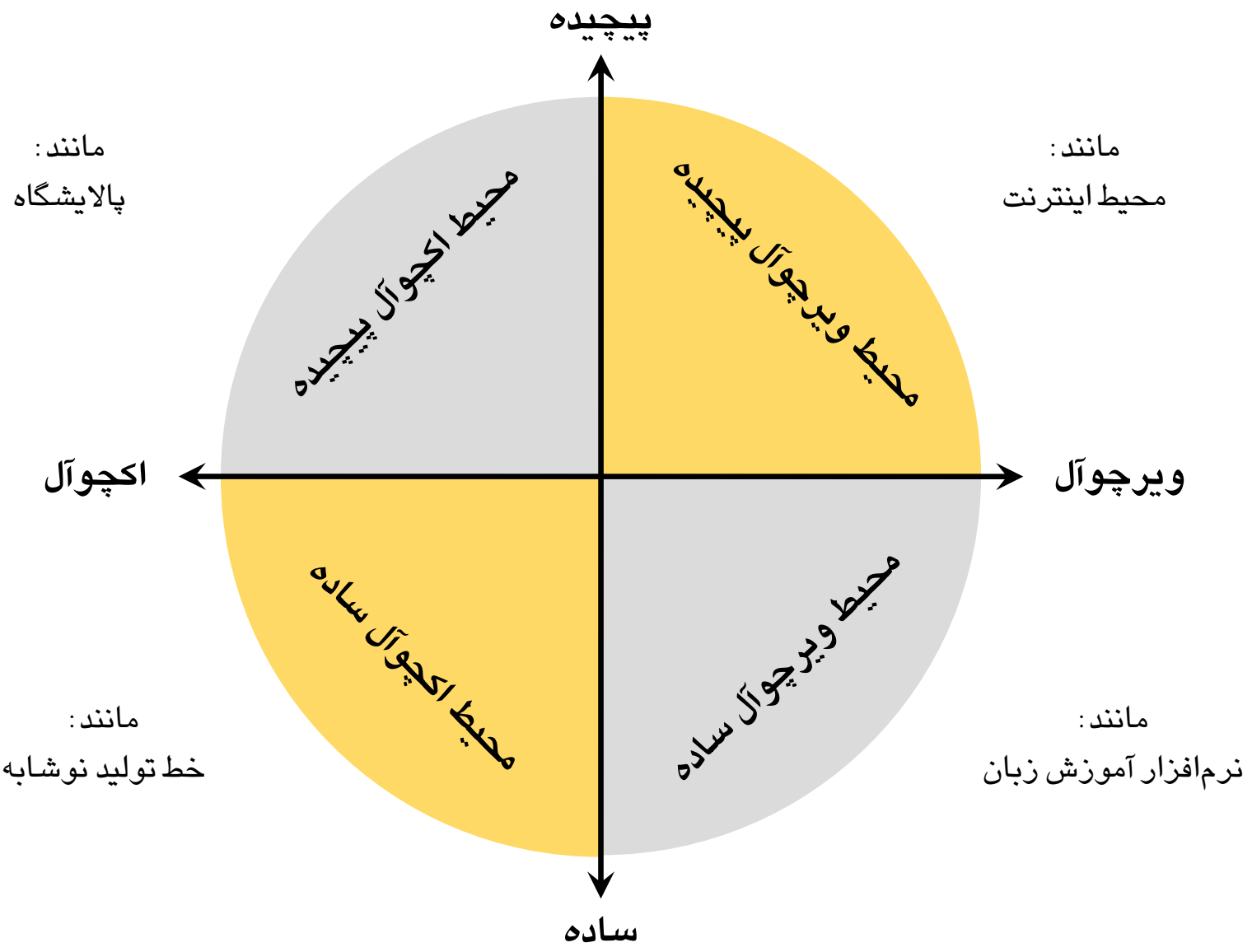
	مشاهده‌پذیر؟	چندعاملی؟	قطعی؟	مقطعی؟	ایستا؟	گسسته؟
بازی شطرنج	کامل	چندعاملی	قطعی	دنباله‌ای	ایستا	گسسته
بازی مار و پله	کامل	چندعاملی	اتفاقی	دنباله‌ای	ایستا	گسسته
خرید اینترنتی	جزئی	تک‌عاملی*	تاحدودی	دنباله‌ای	نیمه‌پویا	گسسته
رانندگی تاکسی	جزئی	چندعاملی	اتفاقی	دنباله‌ای	پویا	پیوسته
تشخیص پزشکی	جزئی	تک‌عاملی	اتفاقی	دنباله‌ای	پویا	پیوسته
کنترلر پالایشگاه	جزئی	تک‌عاملی	اتفاقی	دنباله‌ای	پویا	پیوسته

نوع محیط، روش طراحی عامل را مشخص می‌کند.

هر مجموعه از روش‌های هوش مصنوعی، برای طراحی عامل در نوع خاصی از محیط مناسب است.

انواع محیط

نسبت محیط ساده / پیچیده با محیط اکچوآل / ویرچوآل



بسم الله الرحمن الرحیم

هوش مصنوعی

درس ۹

عامل‌های هوشمند (۳)

Intelligent Agents (3)

کاظم فولادی قلعه
دانشکده مهندسی، دانشکدگان فارابی
دانشگاه تهران

http://courses.fouladi.ir/ai

هوش مصنوعی

عامل‌های هوشمند

۴

ساختار عامل‌ها

ساختار عامل‌ها

THE STRUCTURE OF AGENTS

پیاده‌سازی تابع عامل:
نگاشت ادراک‌ها به کنش‌ها

وظیفه‌ی هوش مصنوعی،
طراحی برنامه‌ی عامل است.

نوعی دستگاه محاسباتی همراه
با حسگرها و کنش‌گرهای
فیزیکی.

وظیفه:
دریافت ادراک‌ها از حسگرها
اجرای برنامه‌ی عامل
ارائه‌ی کنش‌های انتخابی برنامه به کنش‌گرها

مثال: کامپیوتر، خودرو رباتیک با
کامپیوتر داخلی، دوربین و دیگر حسگرها

برنامه‌ی عامل باید متناسب با معماری عامل باشد.

برنامه‌ی عامل مبتنی بر جدول

TABLE-DRIVEN-AGENT

```
function TABLE-DRIVEN-AGENT(percept) returns an action
  persistent: percepts, a sequence, initially empty
              table, a table of actions, indexed by percept sequences, initially fully specied

  append percept to the end of percepts
  action ←LOOKUP(percepts, table)
  return action
```

برنامه‌ی عامل مبتنی بر جدول:

- برای هر ادراک یک بار فراخوانی می‌شود.
- تاریخچه‌ی ادراکی را به روز می‌کند.
- برای تعیین کنش بعدی، از تاریخچه‌ی ادراکی به عنوان اندیس جدول مراجعه استفاده می‌کند.
- کنش مرتبط را برمی‌گرداند.

table

Percept sequence	Action

برنامه‌ی عامل مبتنی بر جدول

مشکلات

TABLE-DRIVEN-AGENT

طول عمر عامل

$$\text{تعداد مدخل‌های جدول} = \sum_{t=1}^{T} |\mathcal{P}|^t$$

اندازه‌ی مجموعه‌ی ادراک‌های عامل

دلایل شکست برنامه‌ی عامل مبتنی بر جدول:

- بزرگ بودن اندازه‌ی جدول (و محدودیت فیزیکی حافظه).
- نبود وقت کافی برای پر کردن جدول توسط طراح.
- عامل نمی‌تواند تمام مدخل‌های جدول را به درستی با تجربه‌اش پر کند.
- طراح هیچ راهنمایی برای پر کردن جدول ندارد.
- عامل حاصل خودمختاری ندارد.

table

Percept sequence	Action

برنامه‌ی عامل مبتنی بر جدول

حل مشکلات: چالش کلیدی هوش مصنوعی

چالش:
نوشتن برنامه‌های کوچک به جای جدول‌های بسیار بزرگ برای تولید رفتار رسیونال

مثال: الگوریتم نیوتن برای محاسبه‌ی جذر
به جای جدول بزرگ جذر اعداد

Percept x	Action z
1.0	1.000000000000000
1.1	1.048808848170152
1.2	1.095445115010332
1.3	1.140175425099138
1.4	1.183215956619923
1.5	1.224744871391589
1.6	1.264911064067352
1.7	1.303840481040530
1.8	1.341640786499874
1.9	1.378404875209022
⋮	⋮

```
function SQRT(x)
  z ← 1.0                          /* initial guess */
  repeat until |z² − x| < 10⁻¹⁵
        z ← z − (z² − x)/(2z)
  end
  return z
```

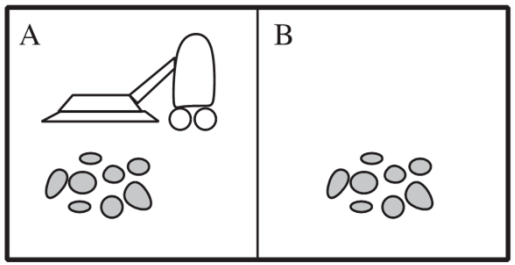
Figure 2.2 Part of the ideal mapping for the square-root problem (accurate to 15 digits), and a corresponding program that implements the ideal mapping.

مثال: دنیای جاروبرقی

برنامه‌ی عامل (در قالب کد)

function REFLEX-VACUUM-AGENT([*location*,*status*]) **returns** an action

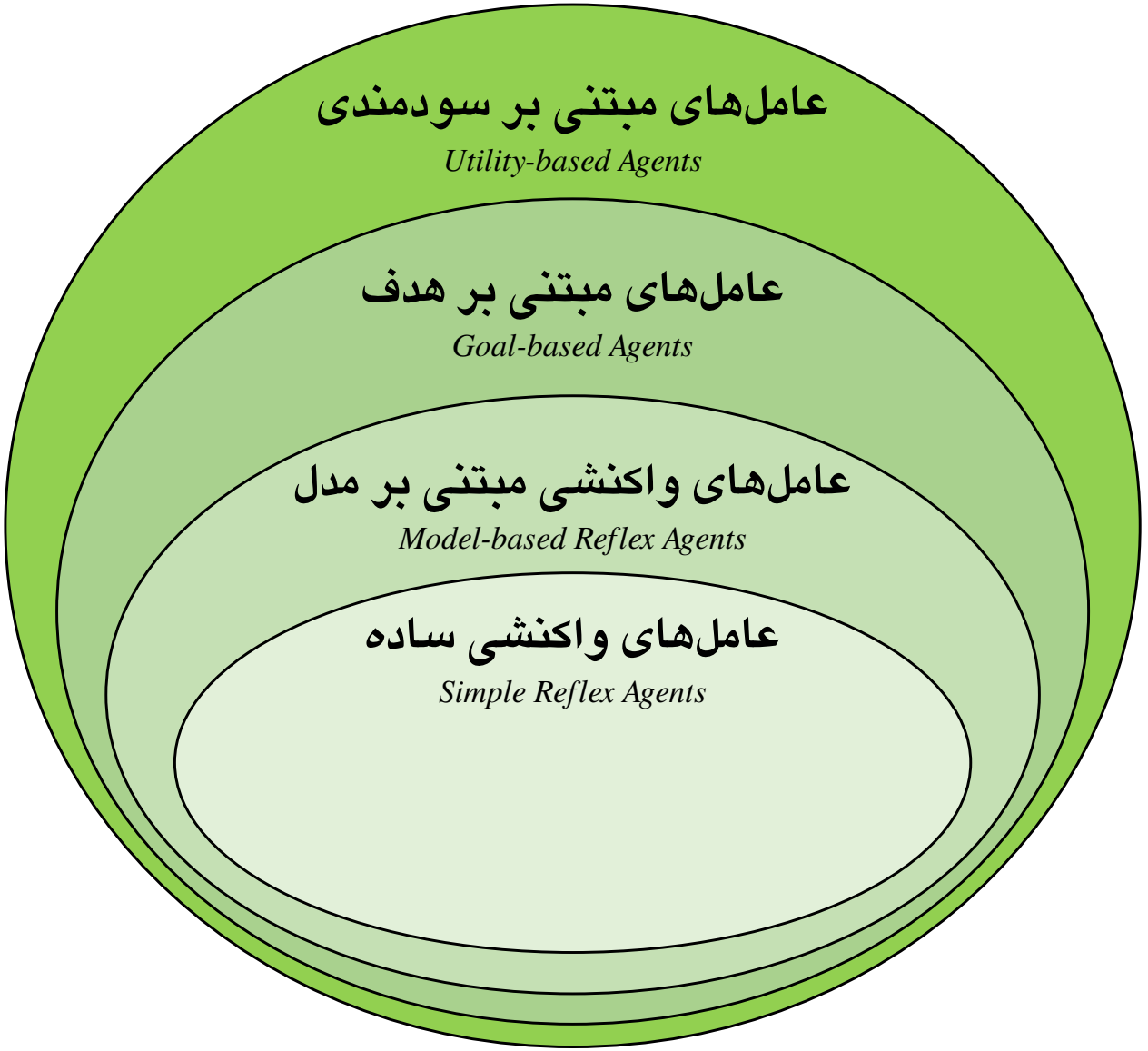
if *status* = *Dirty* **then return** *Suck*
else if *location* = *A* **then return** *Right*
else if *location* = *B* **then return** *Left*



Percept sequence	Action
[*A*, *Clean*]	*Right*
[*A*, *Dirty*]	*Suck*
[*B*, *Clean*]	*Left*
[*B*, *Dirty*]	*Suck*
[*A*, *Clean*], [*A*, *Clean*]	*Right*
[*A*, *Clean*], [*A*, *Dirty*]	*Suck*
⋮	⋮
[*A*, *Clean*], [*A*, *Clean*], [*A*, *Clean*]	*Right*
[*A*, *Clean*], [*A*, *Clean*], [*A*, *Dirty*]	*Suck*
⋮	⋮

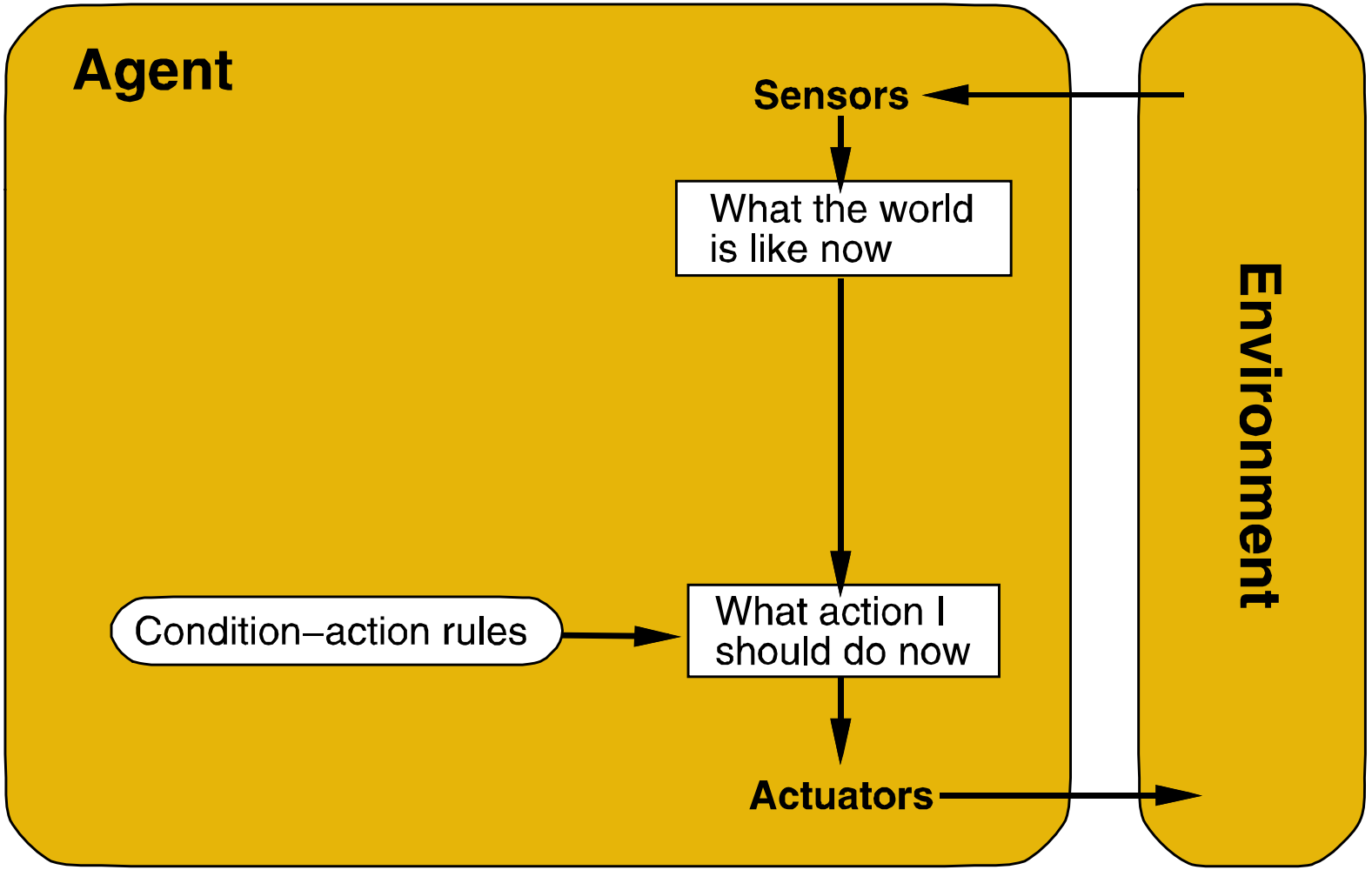
Figure 2.3 Partial tabulation of a simple agent function for the vacuum-cleaner world shown in Figure 2.2.

ساختار برنامه‌های عامل



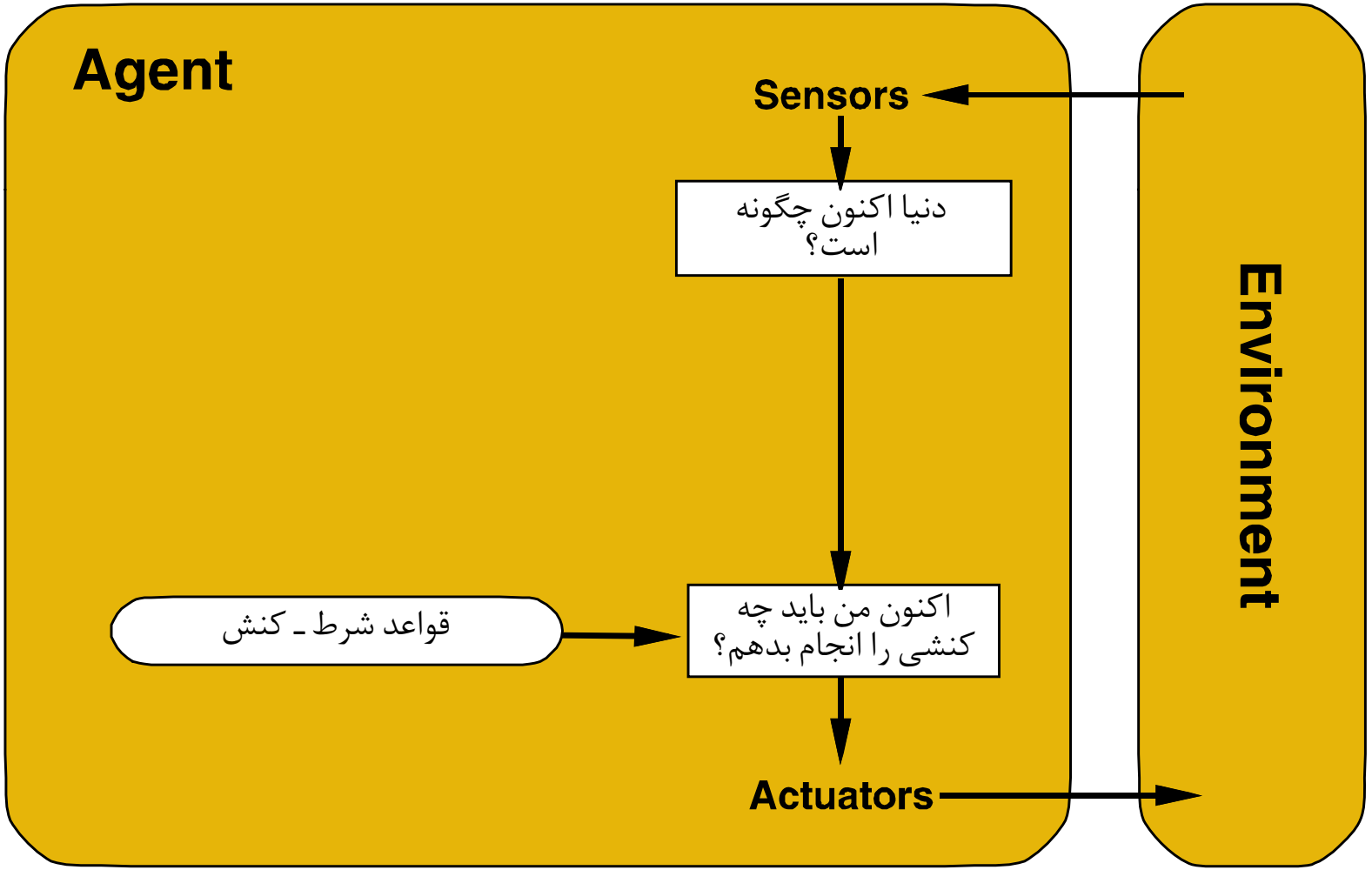
عامل واکنشی ساده

SIMPLE REFLEX AGENT



عامل واکنشی ساده

SIMPLE REFLEX AGENT



عامل واکنشی ساده

SIMPLE REFLEX AGENT

ساده‌ترین نوع عامل

کنش‌ها را بر اساس ادراک فعلی انتخاب می‌کند و تاریخچه‌ی ادراکی را نادیده می‌گیرد.

انتخاب کنش بر مبنای قواعد شرط ـ کنش

condition-action rules
situation-action rules
if-then rules

مثال: **اگر** خودروی جلویی در حال ترمز کردن است **آن‌گاه** شروع به ترمز گرفتن کن

مزایا	معایب
سادگی بالا	حوزه‌ی کاربرد محدود
پیاده‌سازی کارآمد و سریع	هوشمندی پایین
نیاز محدود به منابع	مناسب فقط برای محیط مشاهده‌پذیر کامل
	امکان گیر افتادن در حلقه‌ی بی‌نهایت در محیط مشاهده‌پذیر جزئی: اجتناب با **تصادفی‌سازی** رفتار

عامل واکنشی ساده

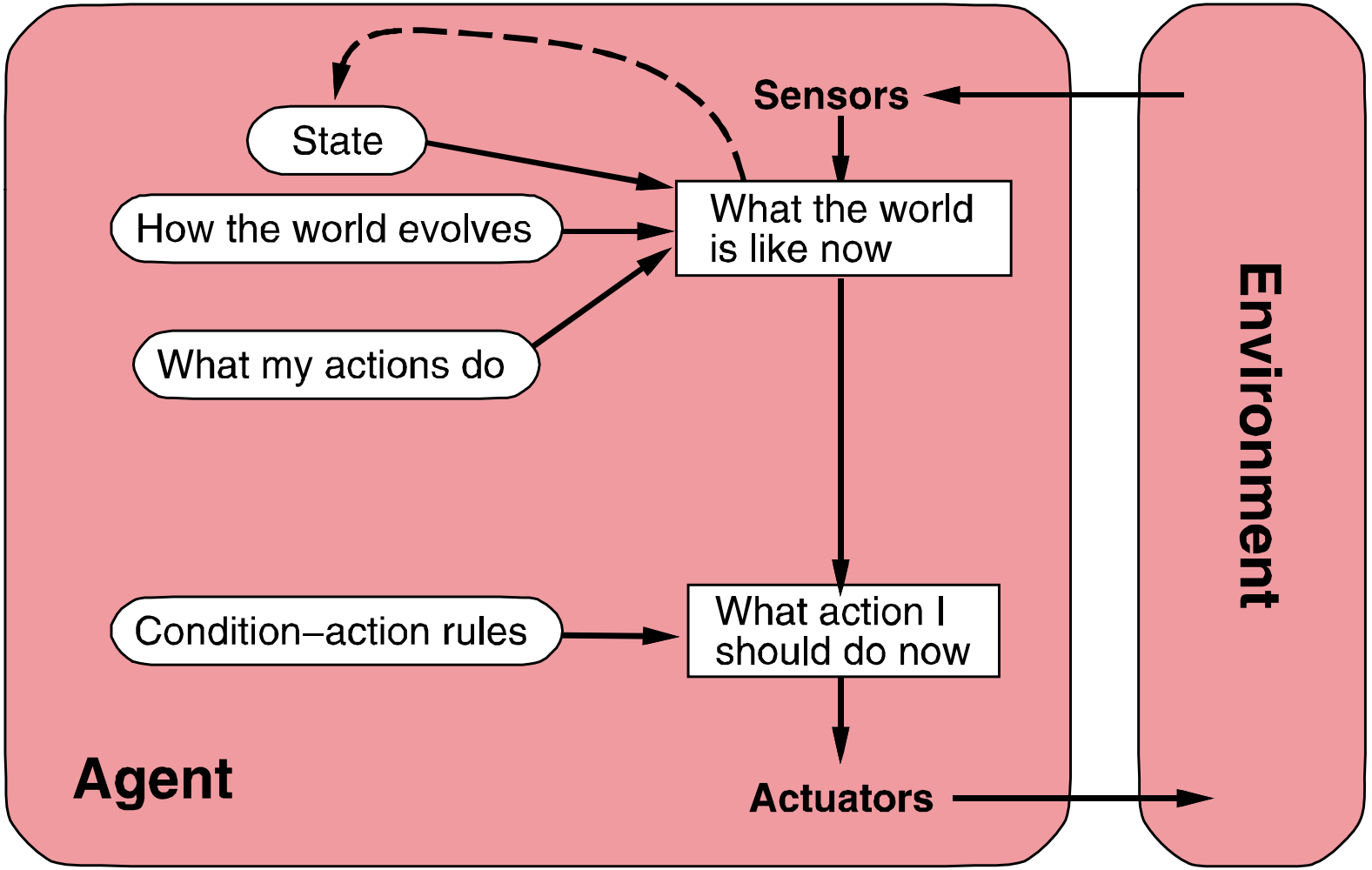
SIMPLE REFLEX AGENT

function SIMPLE-REFLEX-AGENT(*percept*) **returns** an action
  persistent: *rules*, a set of conditionaction rules

  state ←I NTERPRET-INPUT(*percept*)
  rule ←R ULE-MATCH(*state*, *rules*)
  action ← *rule*.ACTION
  return *action*

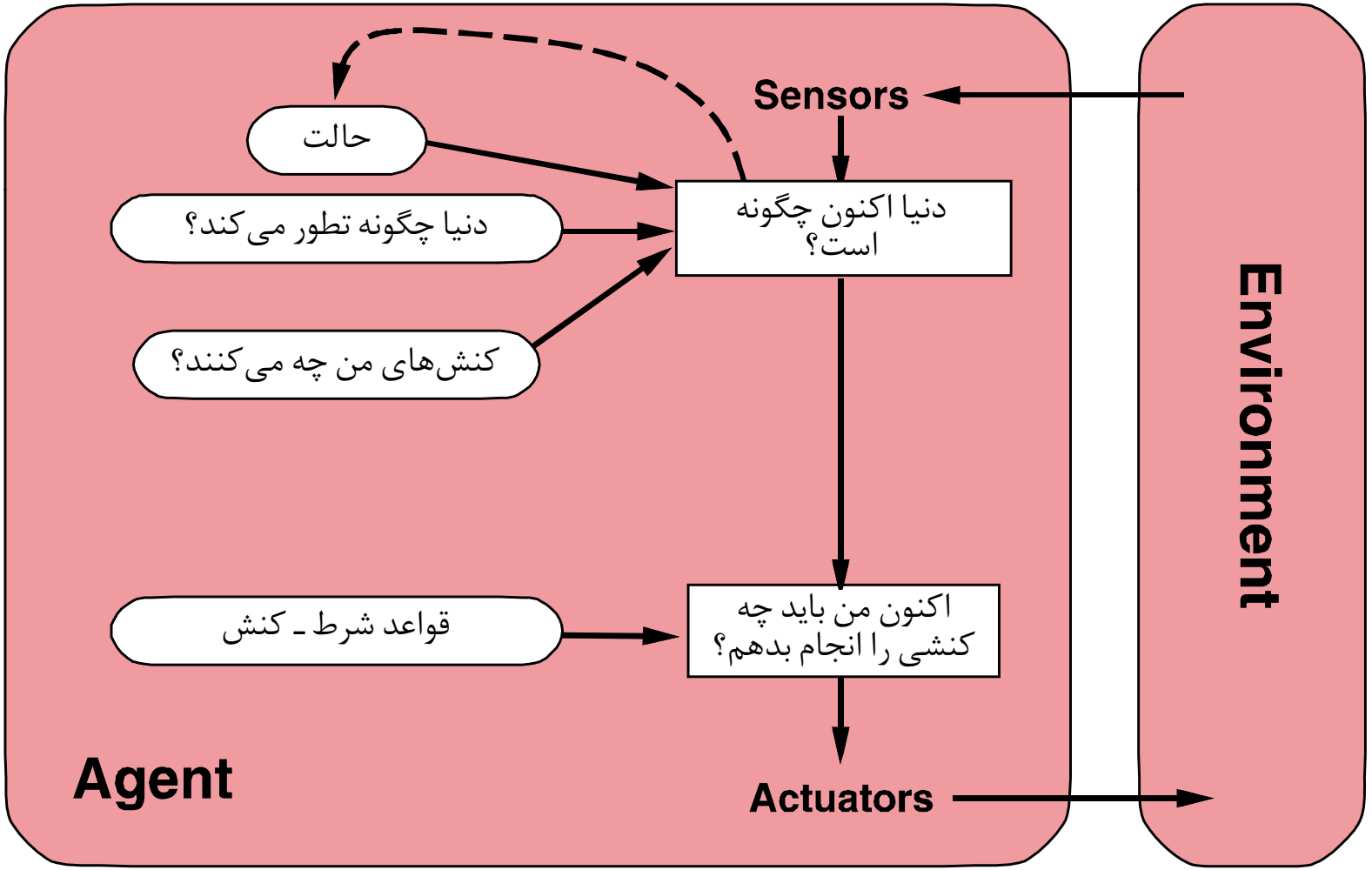
عامل واکنشی مبتنی بر مدل (واکنشی با حالت داخلی)

MODEL-BASED REFLEX AGENT (SIMPLE REFLEX WITH STATE AGENT)



عامل واکنشی مبتنی بر مدل (واکنشی با حالت داخلی)

MODEL-BASED REFLEX AGENT (SIMPLE REFLEX WITH STATE AGENT)



عامل واکنشی مبتنی بر مدل (واکنشی با حالت داخلی)

MODEL-BASED REFLEX AGENT (SIMPLE REFLEX WITH STATE AGENT)

عامل، از حالت و مدل دنیای پیرامونش
برای دنبال کردن بخش‌هایی از دنیا که همیشه نمی‌تواند ببیند، استفاده می‌کند.

مدل: شامل دو نوع اطلاعات:
(۱) چگونگی تطور دنیا مستقل از خود عامل (مدل دنیا: ساده / پیچیده)
(۲) چگونگی اثر کنش‌های عامل بر دنیا

مثال حالت: نگهداری فریم قبلی تصویر چراغ خودوری جلویی

عامل واکنشی مبتنی بر مدل (واکنشی با حالت داخلی)

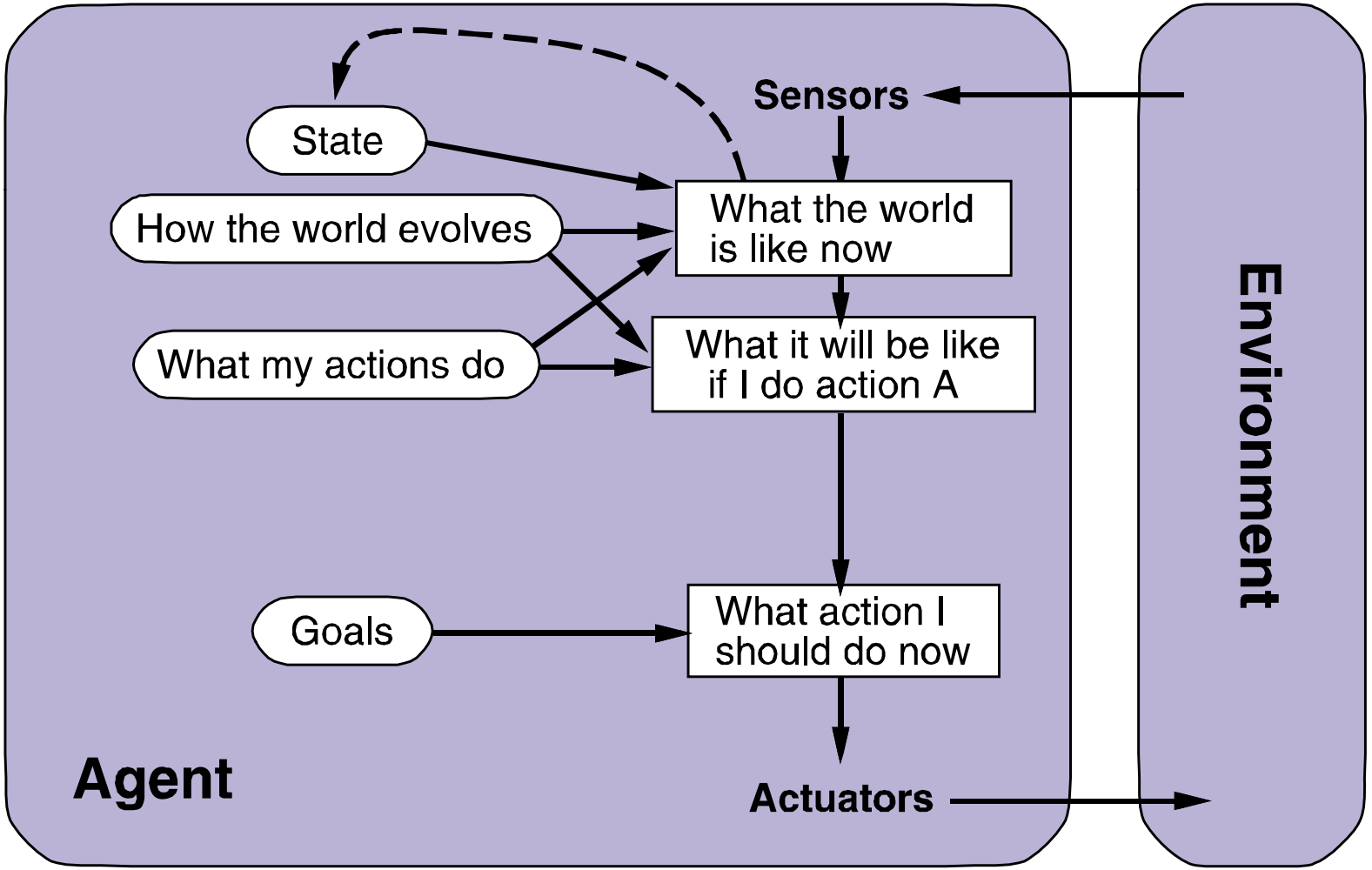
MODEL-BASED REFLEX AGENT (SIMPLE REFLEX WITH STATE AGENT)

function MODEL-BASED-REFLEX-AGENT(*percept*) **returns** an action
  persistent: *state*, the agents current conception of the world state
    model, a description of how the next state depends on current state and action
    rules, a set of conditionaction rules
    action, the most recent action, initially none

  state ←UPDATE -STATE(*state*, *action*, *percept*, *model*)
  rule ←RULE -MATCH(*state*, *rules*)
  action ← *rule*.ACTION
  return *action*

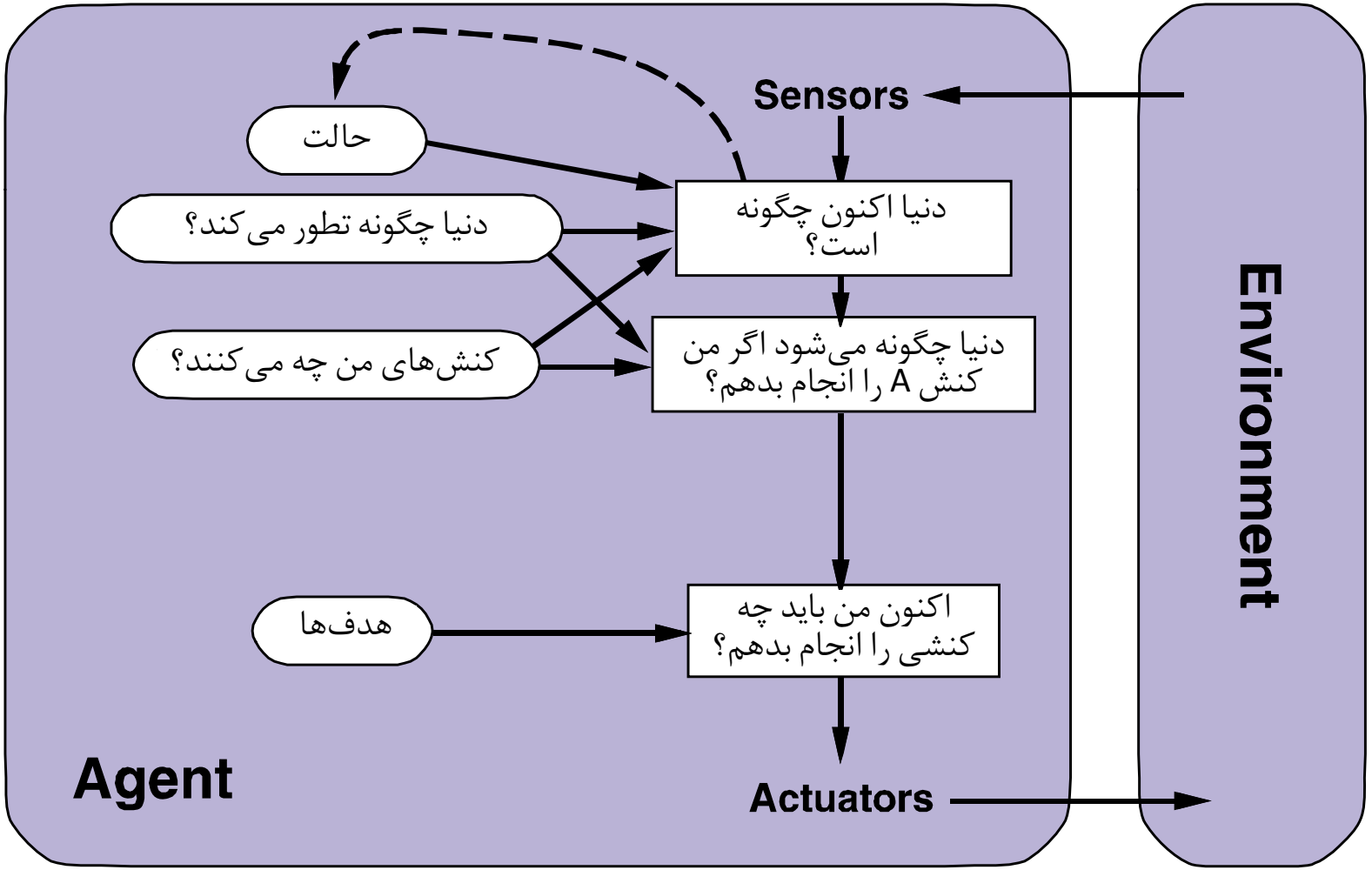
عامل مبتنی بر هدف

GOAL-BASED AGENTS



عامل مبتنی بر هدف

GOAL-BASED AGENTS



عامل مبتنی بر هدف

GOAL-BASED AGENTS

اطلاع از حالت فعلی محیط همیشه برای تصمیم‌گیری در مورد کنش بعدی کافی نیست.
عامل علاوه بر حالت فعلی، به نوعی **اطلاعات در مورد هدف** (حالت مطلوب) نیاز دارد.

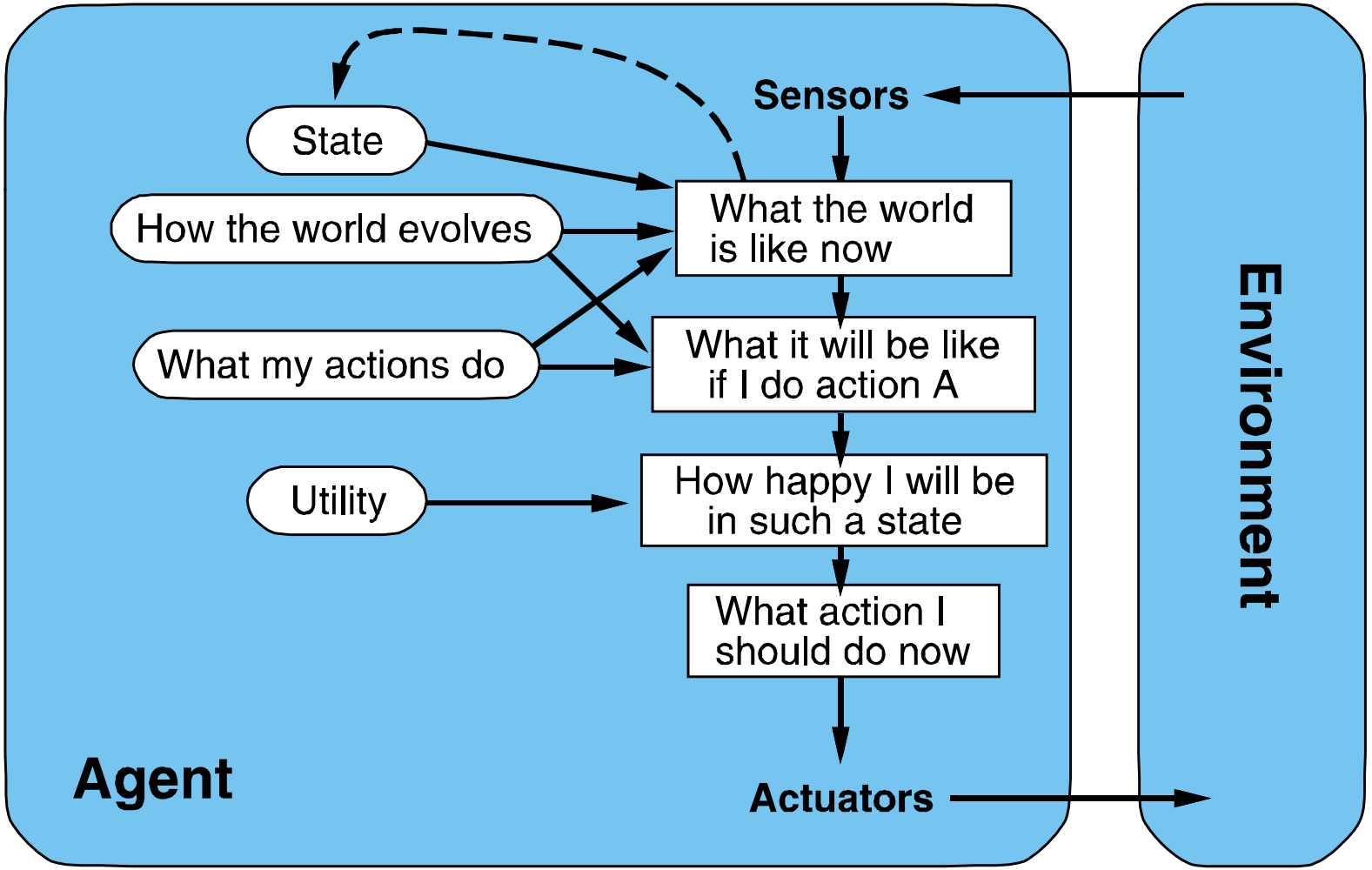
گام‌های رسیدن به هدف

(۱) در یک گام: یک کنش سرراست
(۲) در چند گام: دنباله‌ی کنش‌های عامل (تکنیک‌های **جستجو**؛ **طرح‌ریزی**)

تصمیم‌گیری عامل مبتنی بر هدف، با عامل واکنشی (با قواعد شرط ـ کنش) اساساً متفاوت است.

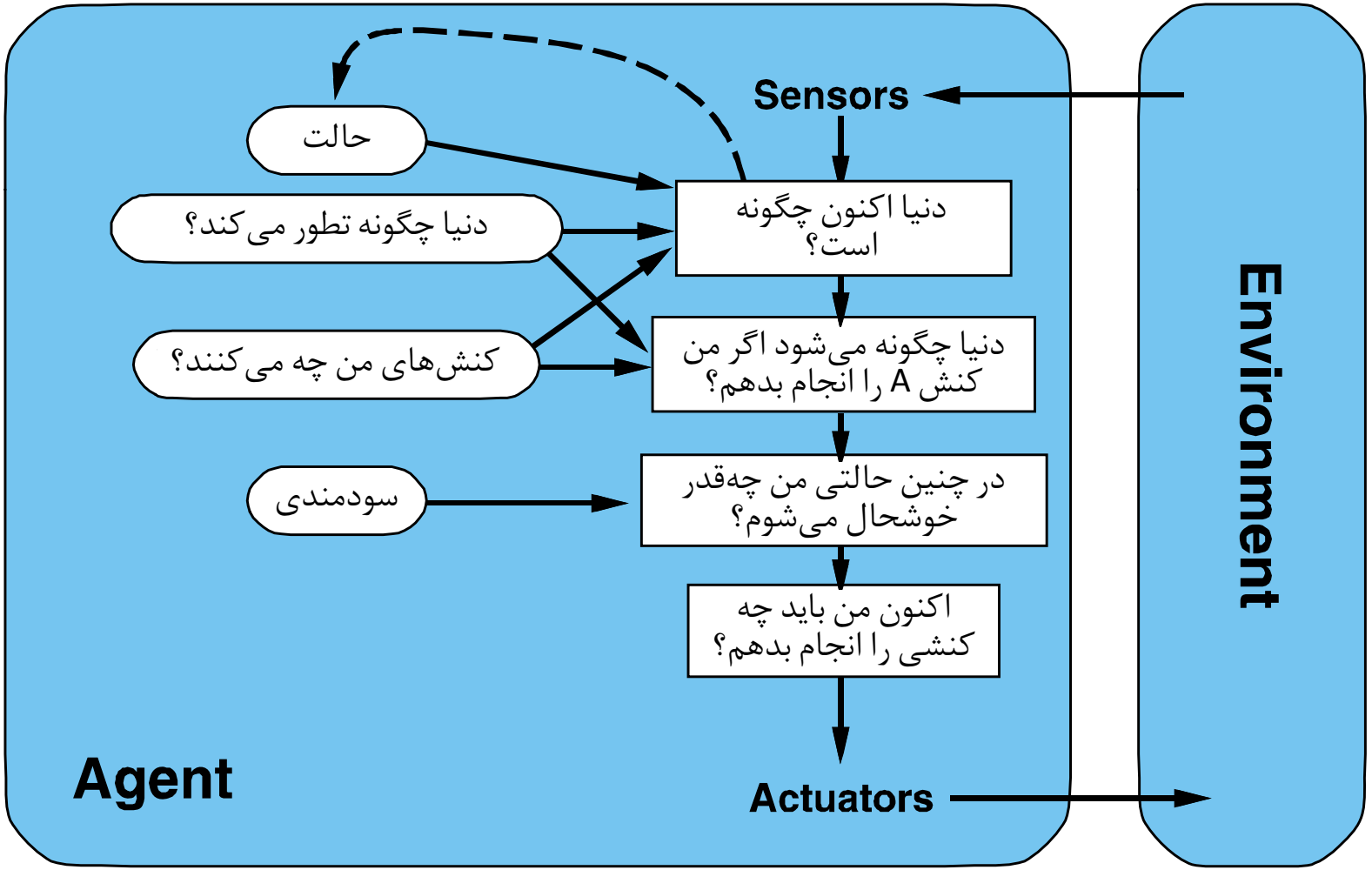
عامل مبتنی بر سودمندی

UTILITY-BASED AGENTS



عامل مبتنی بر سودمندی

UTILITY-BASED AGENTS



عامل مبتنی بر سودمندی

UTILITY-BASED AGENTS

اطلاعات در مورد هدف برای دستیابی به **کارآیی بهینه** کافی نیست.
هدف انتخاب دنباله‌ای از کنش‌هاست که برای عامل **سودمندی** بیشتری دارد.

مثل سریع‌تر، امن‌تر، قابل اطمینان‌تر، ارزان‌تر، ... در مسئله‌ی تاکسی هوشمند.

استفاده از تابع سودمندی برای تصمیم‌گیری رسیونال در شرایط

(۱) وجود اهداف متناقض: با ایجاد بده–بستان بین آنها (مثل سرعت و امنیت)

(۲) وجود اهداف چندگانه: عامل می‌توان قصد رسیدن به آنها را داشته باشد ولی هیچ‌کدام برایش قطعیت ندارد.
امکان سنجش شانس موفقیت‌ها به میزان اهمیت اهداف.

عامل مبتنی بر سودمندی

تابع سودمندی

UTILITY FUNCTION

تابع سودمندی

(۱) حالت (دنباله‌ی حالت‌های محیط) را به یک عدد حقیقی نگاشت می‌دهد.

$$u : S^* \rightarrow \mathbb{R}$$

(۲) تناقض‌ها را از طریق بده–بستان (trade-off) رفع می‌کند.

(۳) عدم اطمینان را از طریق ارائه‌ی معیاری برای شانس موفقیت رفع می‌کند.

هر عاملی که یک تابع سودمندی صریح داشته باشد، می‌تواند تصمیم‌های رسیونال بگیرد.
عامل رسیونال تلاش می‌کند که مقدار متوسط تابع سودمندی خودش را ماکزیمم کند.

نسبت میان تابع سودمندی و معیار کارآیی

تابع سودمندی *Utility Function*	معیار کارآیی *Performance Measure*
در اختیار عامل	در اختیار محیط
درونی	بیرونی
دارای دید محلی (زمانی ـ مکانی)	دارای دید سراسری (زمانی ـ مکانی)

موفقیت عامل به هم‌جهت بودن تابع سودمندی با معیار کارآیی بستگی دارد.

عامل‌های یادگیرنده

یادگیری

LEARNING AGENTS

یادگیری به یک عامل اجازه می‌دهد که در **محیط‌های ابتدائاً ناشناخته** عمل کند و سپس از آنچه دانایی اولیه‌اش به‌تنهایی ممکن بود اجازه بدهد، **شایسته‌تر** شود.

ساخت عامل‌های هوش مصنوعی	
ایجاد مؤلفه‌ی یادگیرنده و آموزش عامل	برنامه‌نویسی از صفر

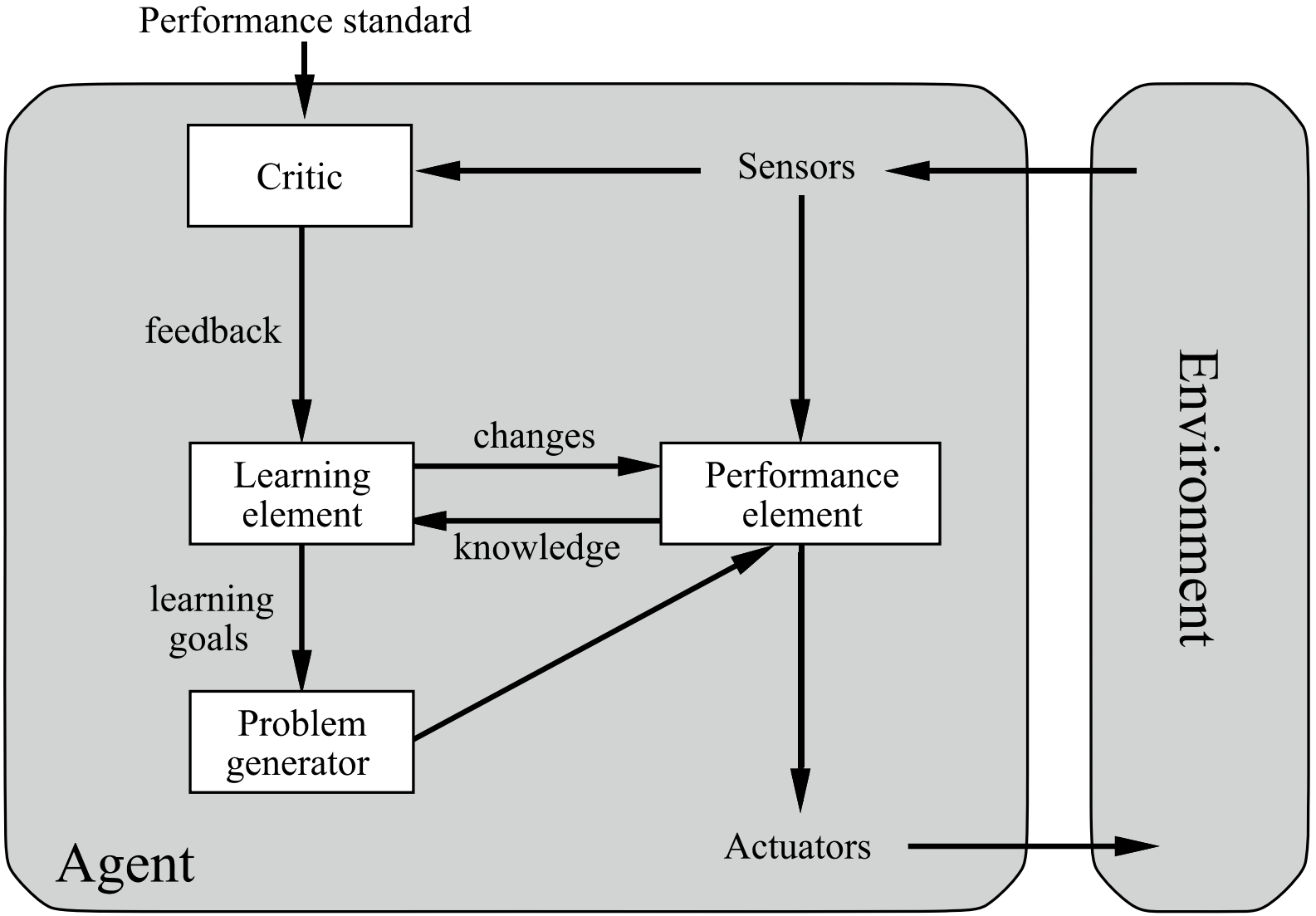
- **پیشنهاد آلن تورینگ (۱۹۵۰)**

دو پرسش برای طراحی عامل یادگیرنده:

(۱) **عنصر انجام‌دهنده چه باشد؟** (یکی از ساختارهای چهارگانه برنامه عامل)
(۲) **چگونه یادگیری انجام شود؟** (روش‌های گوناگون و مؤلفه‌های مختلف ساختار عامل)

عامل‌های یادگیرنده

LEARNING AGENTS



عامل‌های یادگیرنده

اجزای داخلی

معیار کارآیی استاندارد: Performance standard

ثابت است.
باید به طور کامل خارج از عامل باشد
(عامل نباید بتواند آن را تغییر دهد).

منتقد
Critic

به عنصر یادگیرنده فیدبک می‌دهد:

با توجه به یک معیار استاندارد کارآیی ثابت به عنصر یادگیرنده می‌گوید که عامل چه‌قدر خوب عمل کرده است.

مسئول انتخاب کنش بیرونی:

می‌تواند هر یک از ساختارهای چهارگانه‌ی برنامه‌ی عامل را داشته باشد.

عنصر انجام‌دهنده
Performance element

عنصر یادگیرنده
Learning element

مسئول بهبود بخشیدن به کارآیی:

تغییرات عنصر انجام‌دهنده را برای بهتر عمل کردن در آینده مشخص می‌کند.

مولد مسئله
Problem generator

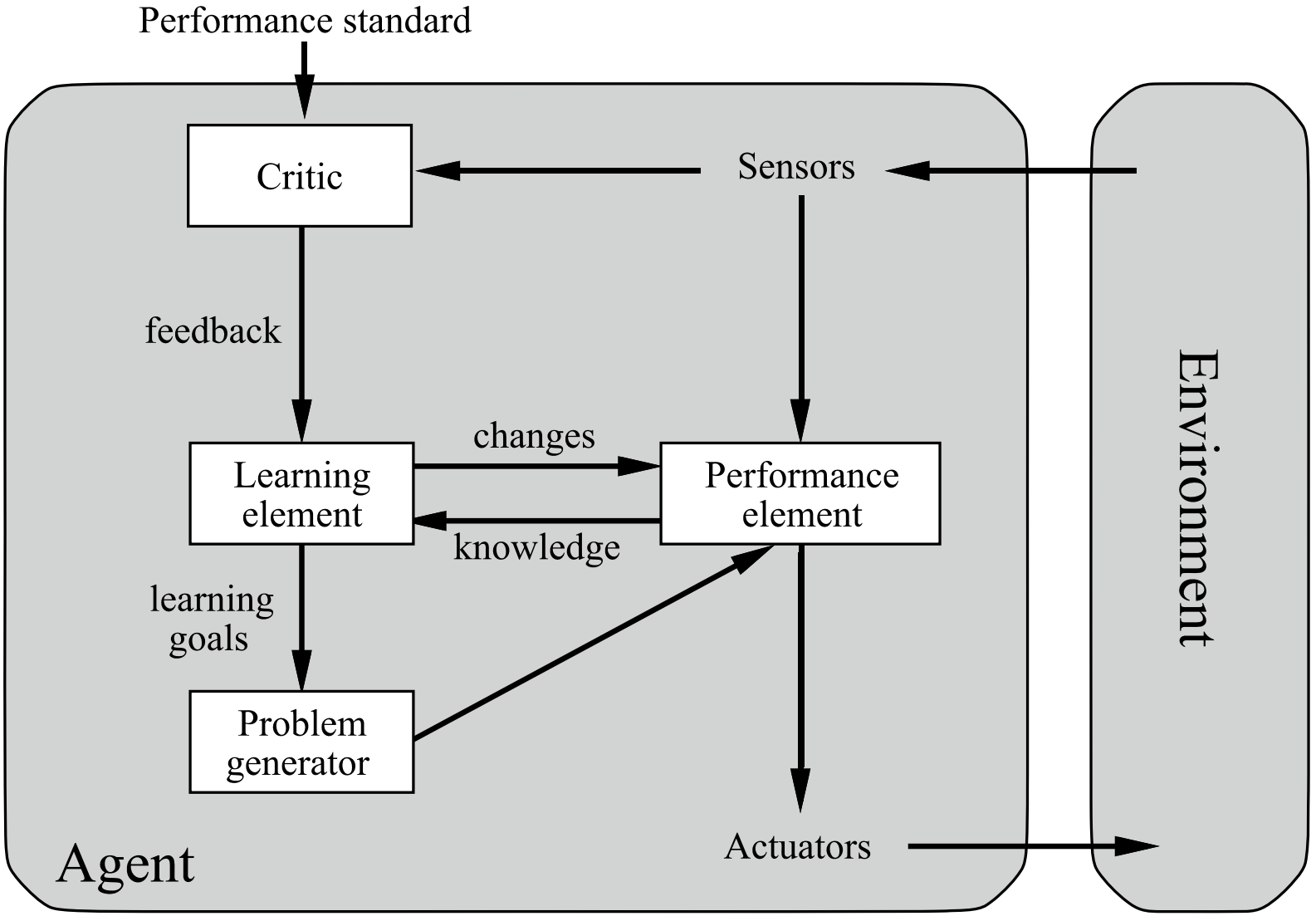
مسئول پیشنهاد کنش‌های منجر به تجربه‌های تازه و آموزنده:

پیشنهاد کنش‌های کاوشگرانه بر اساس اهداف یادگیری

یادگیری در عامل هوشمند: فرآیند تغییر هر جزء عامل به این منظور که آن جزء تطابق بیشتری با اطلاعات فیدبک داشته باشد و از این طریق کارآیی کل عامل بهبود یابد.

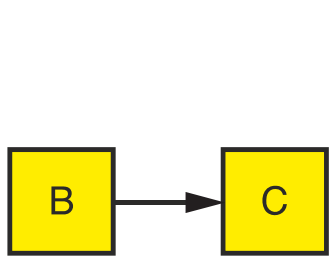
عامل‌های یادگیرنده

LEARNING AGENTS



نحوه‌ی کار اجزای برنامه‌های عامل

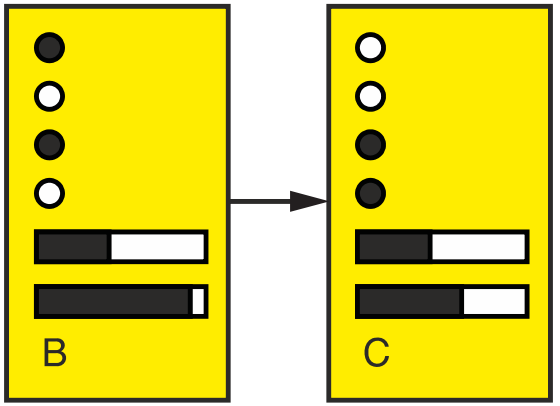
سه روش برای بازنمایی حالت‌ها و گذار بین آنها



(a) Atomic

یک حالت یک جعبه‌ی سیاه بدون ساختار داخلی است.

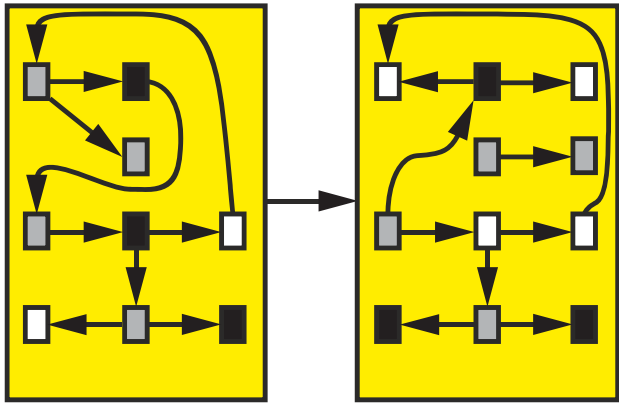
- Search
- Game-Playing
- Hidden Markov Models
- Markov Decision Process



(b) Factored

یک حالت از برداری از مقادیر خصیصه‌ها تشکیل شده است (مقادیر بولی، عدد حقیقی، لیترال).

- Constraint Satisfaction
- Propositional Logic
- Planning
- Bayesian Networks
- Machine Learning



(c) Structured

یک حالت شامل اشیایی است که می‌توانند خصیصه‌های خود و روابط با دیگر اشیا را داشته باشند.

- Relational Databases
- First-Order Logic
- First-Order Probability Models
- Knowledge-Based Learning
- Natural Language Processing

Increase in Expressiveness افزایش رسا بودن

نحوه‌ی کار اجزای برنامه‌های عامل

از نظر نگاشت مفاهیم به مکان‌ها در حافظه‌ی فیزیکی (در یک کامپیوتر یا در یک مغز)

بازنمایی محلی‌گرا
Localist Representation

وجود نگاشت یک به یک میان مفاهیم و مکان‌های حافظه

نگاشت از مفهوم به مکان حافظه به‌صورت مطلق است.

تغییر در چند بیت محدود،
باعث تولید مفهوم دیگری می‌شود
که معنای کاملاً متفاوتی دارد.

بازنمایی توزیع‌شده
Distributed Representation

بازنمایی یک مفهوم بر روی تعداد زیادی مکان حافظه پخش می‌شود و هر مکان حافظه به‌عنوان یک بخشی از بازنمایی مفاهیم مختلف چندگانه به‌کارگرفته می‌شود.

مقاوم‌تر در برابر نویز و فقدان اطلاعات

هر مفهوم یک نقطه در فضای چندبعدی است.

تغییر در چند بیت محدود،
باعث جابه‌جایی به نقطه‌ای نزدیک در آن فضا می‌شود
که معنای مشابهی دارد.

برنامه‌ی محیط

ENVIRONMENT PROGRAM

```
procedure RUN-ENVIRONMENT(state, UPDATE-FN, agents, termination)
  inputs: state, the initial state of the environment
          UPDATE-FN, function to modify the environment
          agents, a set of agents
          termination, a predicate to test when we are done

  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)
```

شبیه‌سازی ارتباط عامل با محیط

برنامه‌ی محیط:

- دادن ورودی ادراکی به هر عامل
- گرفتن کنش هر عامل
- به‌روزرسانی محیط

برنامه‌ی محیط

به همراه ارزیابی عامل‌ها

```
function RUN-EVAL-ENVIRONMENT(state, UPDATE-FN, agents,
                              termination, PERFORMANCE-FN) returns scores
  local variables: scores, a vector the same size as agents, all 0

  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)
    scores ← PERFORMANCE-FN(scores, agents, state)
  until termination(state)
  return scores                                          /* change */
```

شبیه‌سازی ارتباط عامل با محیط

برنامه‌ی محیط:

- دادن ورودی ادراکی به هر عامل
- گرفتن کنش هر عامل
- به‌روزرسانی محیط

+ نگهداری امتیاز کارآیی بر هر عامل

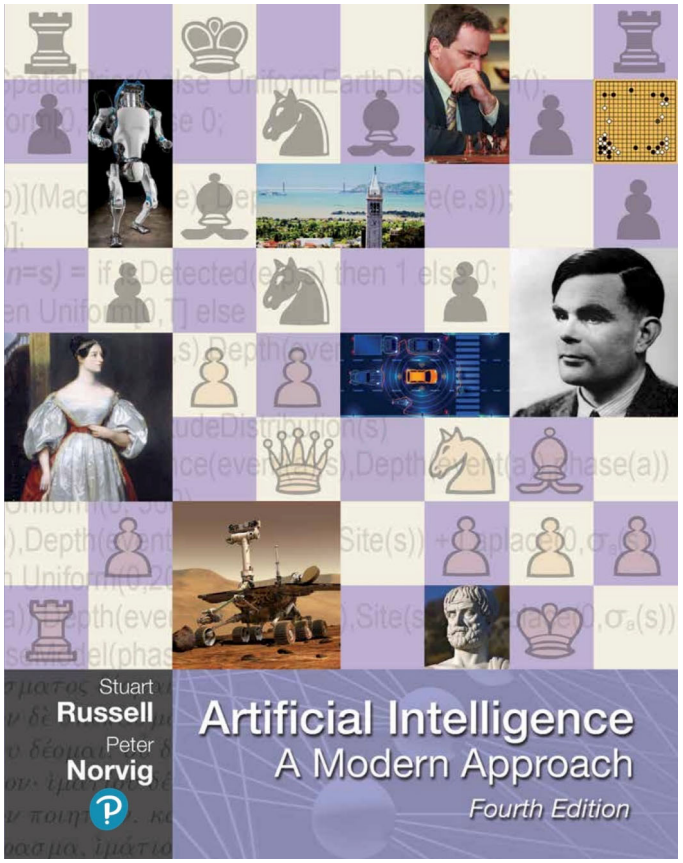
هوش مصنوعی

عامل‌های هوشمند

۵

منابع، مطالعه، تکلیف

منبع اصلی



Stuart Russell and Peter Norvig,
Artificial Intelligence: A Modern Approach,
4th Edition, Prentice Hall, 2020.

Chapter 2

CHAPTER 2

INTELLIGENT AGENTS

In which we discuss the nature of agents, perfect or otherwise, the diversity of environments, and the resulting menagerie of agent types.

Chapter 1 identified the concept of **rational agents** as central to our approach to artificial intelligence. In this chapter, we make this notion more concrete. We will see that the concept of rationality can be applied to a wide variety of agents operating in any imaginable environment. Our plan in this book is to use this concept to develop a small set of design principles for building successful agents—systems that can reasonably be called **intelligent**.

We begin by examining agents, environments, and the coupling between them. The observation that some agents behave better than others leads naturally to the idea of a rational agent—one that behaves as well as possible. How well an agent can behave depends on the nature of the environment; some environments are more difficult than others. We give a crude categorization of environments and show how properties of an environment influence the design of suitable agents for that environment. We describe a number of basic "skeleton" agent designs, which we flesh out in the rest of the book.

2.1 Agents and Environments

Environment
Sensor
Actuator

An **agent** is anything that can be viewed as perceiving its **environment** through **sensors** and acting upon that environment through **actuators**. This simple idea is illustrated in Figure 2.1. A human agent has eyes, ears, and other organs for sensors and hands, legs, vocal tract, and so on for actuators. A robotic agent might have cameras and infrared range finders for sensors and various motors for actuators. A software agent receives file contents, network packets, and human input (keyboard/mouse/touchscreen/voice) as sensory inputs and acts on the environment by writing files, sending network packets, and displaying information or generating sounds. The environment could be everything—the entire universe! In practice it is just that part of the universe whose state we care about when designing this agent—the part that affects what the agent perceives and that is affected by the agent's actions.

Percept
Percept sequence

We use the term **percept** to refer to the content an agent's sensors are perceiving. An agent's **percept sequence** is the complete history of everything the agent has ever perceived. In general, *an agent's choice of action at any given instant can depend on its built-in knowledge and on the entire percept sequence observed to date, but not on anything it hasn't perceived.* By specifying the agent's choice of action for every possible percept sequence, we have said more or less everything there is to say about the agent. Mathematically speaking, we say that an agent's behavior is described by the **agent function** that maps any given percept sequence to an action.

Agent function