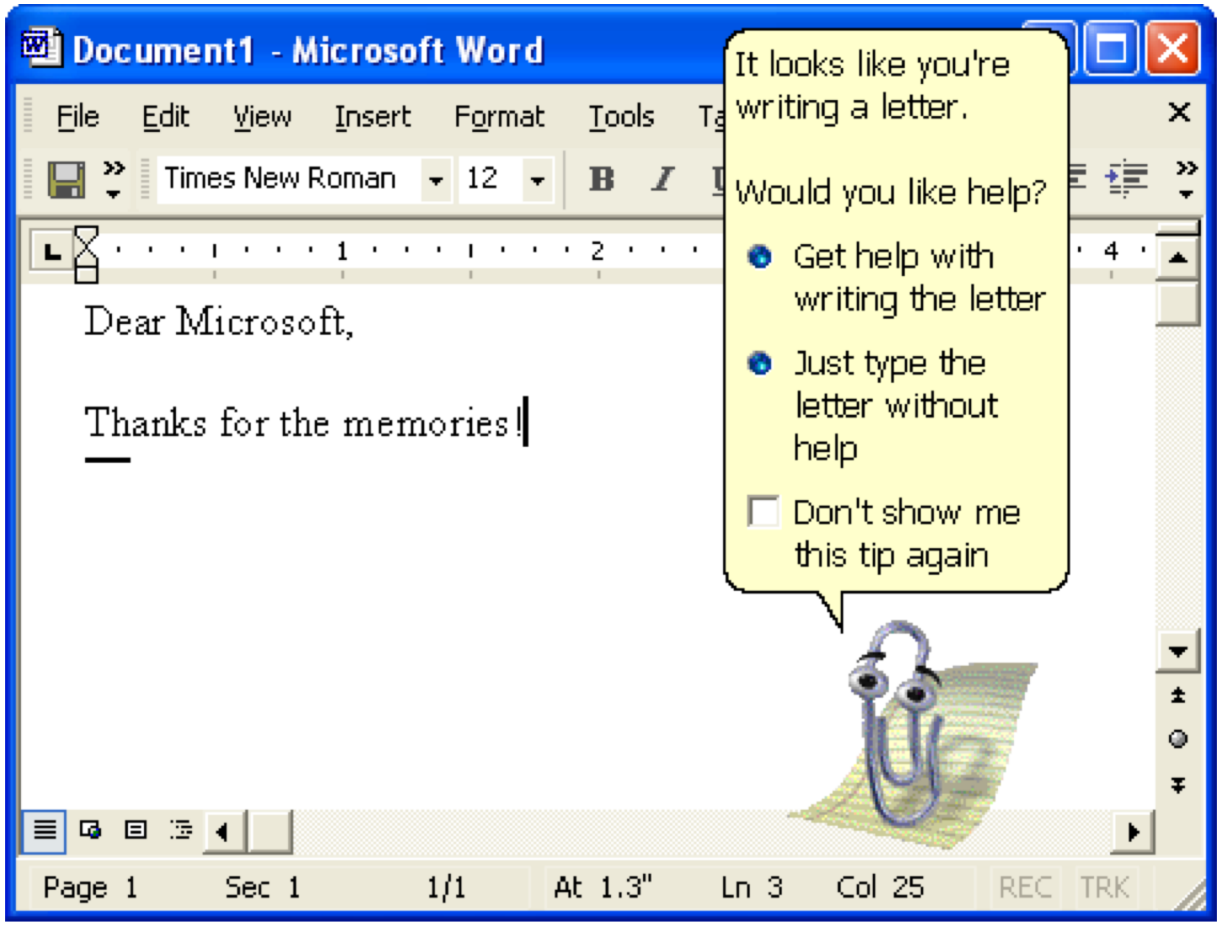
CS 380: Artificial Intelligence

Lecture 2:
Rational Agents

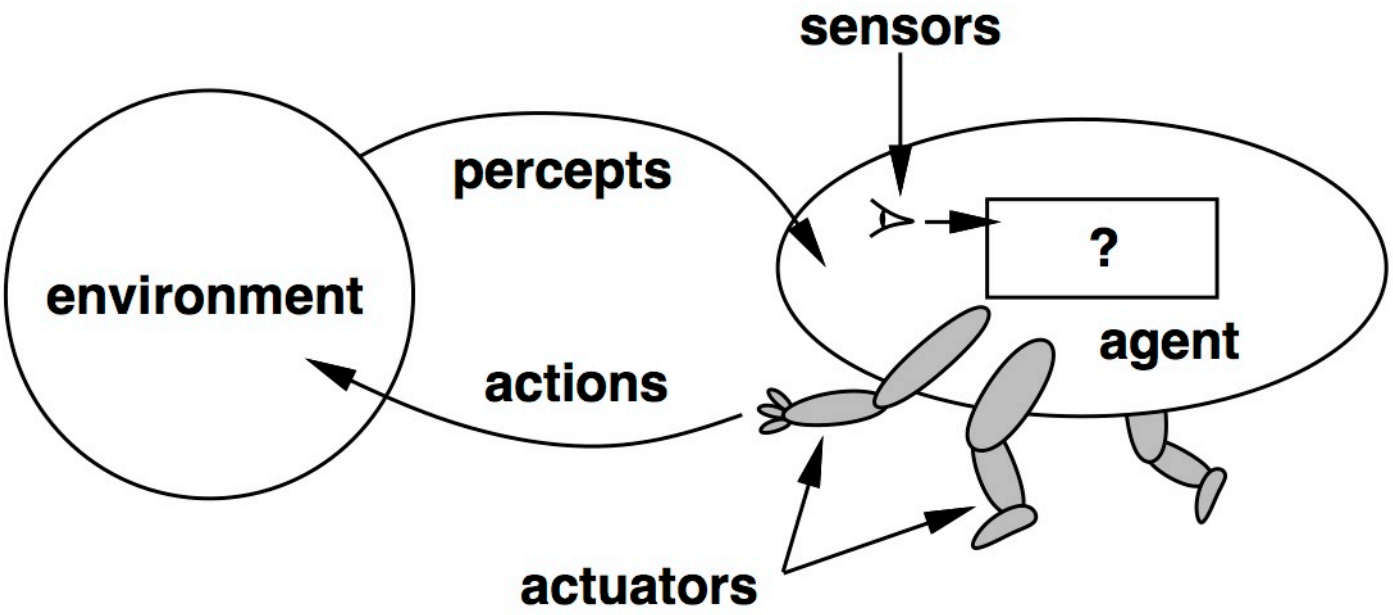
What is an Agent?

- **Agent**: an autonomous entity which observes and acts upon an environment and directs its activity towards achieving goals
- Examples:
  - Robots
  - Software agents
  - ATMs, ticket kiosks, etc.
  - Humans?

What is an Agent?

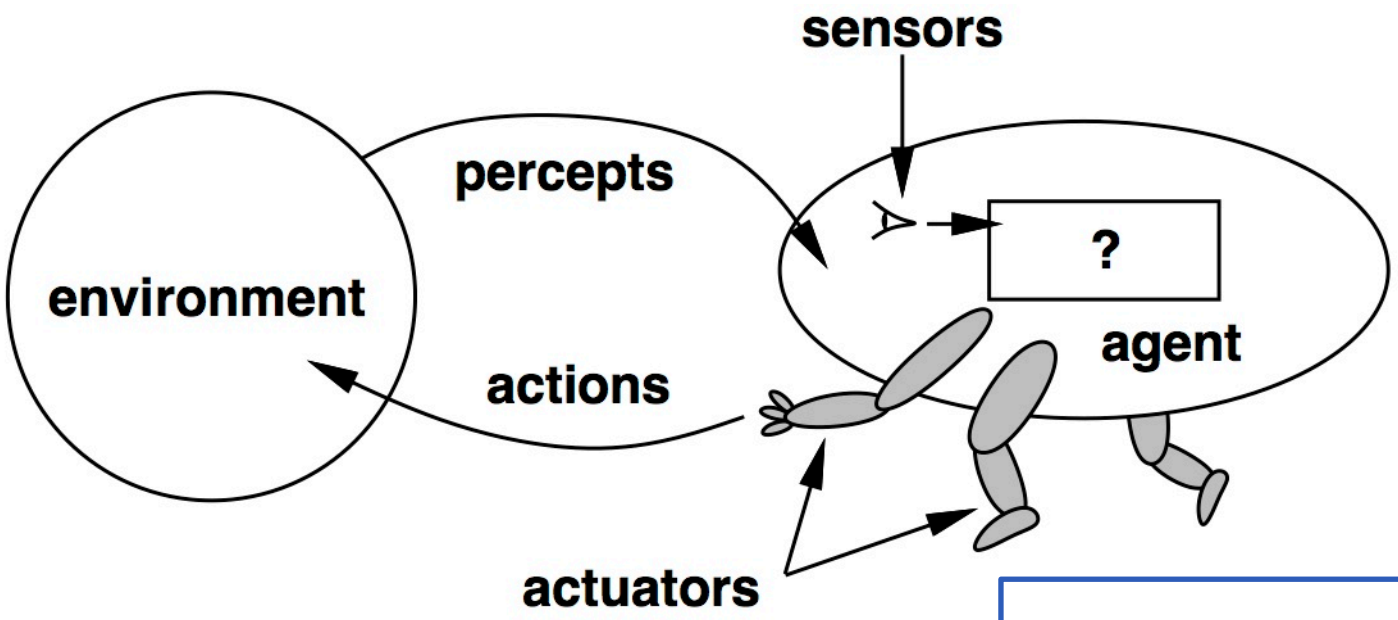


What is an Agent?



- Theoretically: $f : P^* \rightarrow A$
- In practice: the agent runs on a physical system (e.g., a computer) to produce f

What is an Agent?



- Theoretically: $f : P^* \rightarrow A$

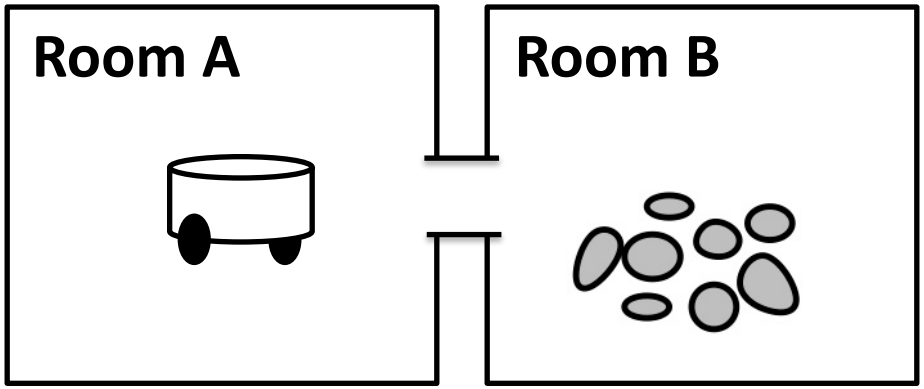
Maps sequences of percepts to actions

- In practice: the agent runs on a physical system (e.g., a computer) to produce f

Percepts

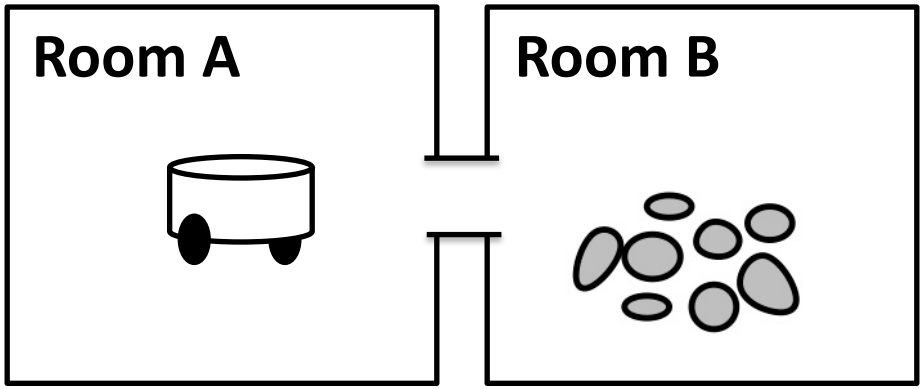
- Can be very simple, can be very complex
- What are the percepts for a self-driving car?
  - < *can go left, can go straight, can go right* >
  - Lane position
  - (above) + distance to lead car
  - (above) + all cars within distance D
  - (above) + road/signal information (e.g., traffic lights)
  - (above) + geospatial navigation information
  - (above) + current/predicted weather
  - … etc.

Example: Vacuum Cleaner



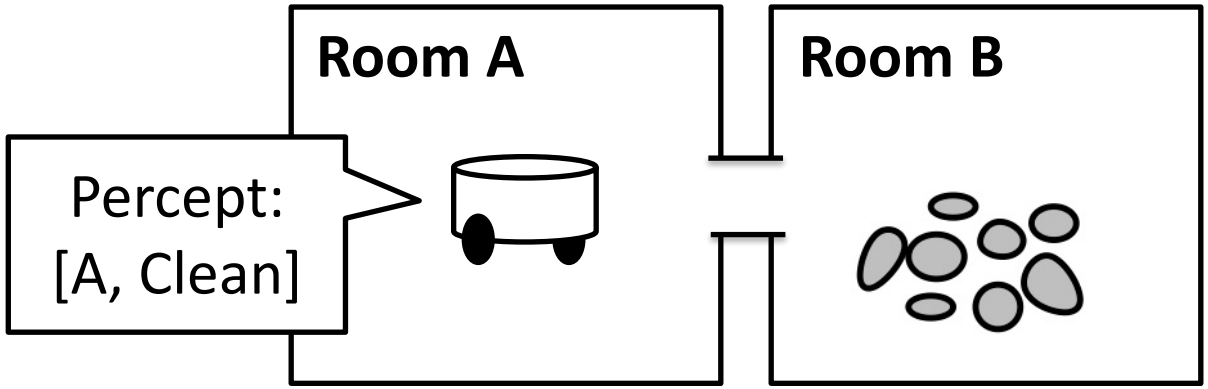
- Percepts
  - Location sensor (Room A or B?)
  - Dirt sensor (Clean or dirty?)
- Actions?

Example: Vacuum Cleaner



- Percepts
  - Location sensor (Room A or B?)
  - Dirt sensor (Clean or dirty?)
- Actions
  - Left, Right, Suck, Wait

Example: Vacuum Cleaner



- Percepts
  - Location sensor (Room A or B?)
  - Dirt sensor (Clean or dirty?)
- Actions
  - Left, Right, Suck, Wait

Example: Vacuum Cleaner

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

Example: Vacuum Cleaner

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

Two questions:

- What is the function?
- Can it be implemented as a computer program?
  - If so, how?

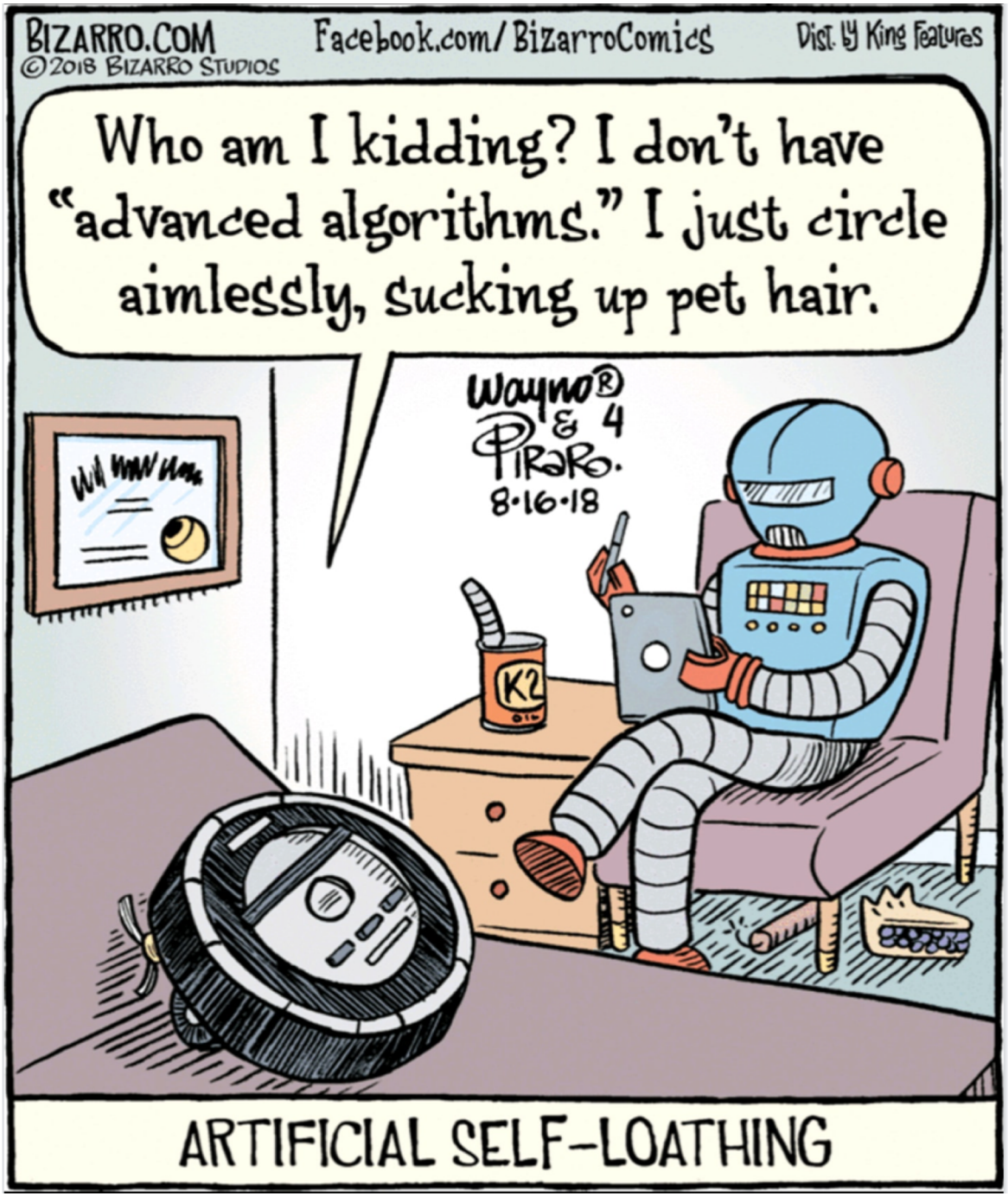
Example: Vacuum Cleaner

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

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*

Facebook.com/BizarroComics
Dist. by King Features



ARTIFICIAL SELF-LOATHING

Rationality

- As we discussed last time, “intelligence” is an ill-defined concept.
- Let us assume a **performance measure** that is given (and usually external) to the agent
  - For example:
    - $1 per room cleaned
    - $10 per room cleaned minus $1 per movement
    - $10 per room cleaned minus $3 per movement
    - etc.
- **Rational agent**:
  - Chooses actions that maximize the expected value of the performance measure given the current percept sequence

Rationality

- Imagine two agents A and B whose performance measure is defined by how much $$ they make at the casino.
  - A and B play one game of blackjack
  - A plays the move that probabilistically has the highest chance of winning
  - B plays a different, riskier move
  - B ends up winning
- Which agent is rational?

Rationality

- Imagine two agents A and B whose performance measure is defined by how much $$ they make at the casino.
  - A and B play one game of blackjack
  - A plays the move that probabilistically has the highest chance of winning
  - B plays a different, riskier move
  - B ends up winning
- Which agent is rational? – **A is the rational agent**
  - Rationality does not imply clairvoyance
  - Rationality does not imply omniscience
  - Thus, rational does **not** necessarily mean successful

How Do We Design a Rational Agent?

- To design a rational agent, we first need to define its environment, performance measure, etc.
- PEAS:
  - Performance
  - Environment
  - Actions
  - Sensors (percepts)

Example (informal): Automated Taxi

- **Performance**:
- **Environment**:
- **Actions**:
- **Sensors**:

Example (informal): Automated Taxi

- **Performance**: profit, safety, legality, …
- **Environment**: streets, pedestrians, cars, weather, traffic lights/signs, …
- **Actions**: steering, throttle, brake, horn, display, …
- **Sensors**: GPS, video, sonar, keyboard, accelerometers, microphones, …

Example: Chess Agent

- **Performance**:
- **Environment**:
- **Actions**:
- **Sensors**:

Example: Chess Agent

- **Performance**:
- **Environment**:
  - Chess board, opponent, chess rules
- **Actions**:
- **Sensors**:

Example: Chess Agent

- **Performance**:
  - Once reached a terminal state (draw, win): -1 for losing, 0 for draw, 1 for winning
- **Environment**:
  - Chess board, opponent, chess rules
- **Actions**:
- **Sensors**:

Example: Chess Agent

- **Performance**:
  - Once reached a terminal state (draw, win): -1 for losing, 0 for draw, 1 for winning
- **Environment**:
  - Chess board, opponent, chess rules
- **Actions**:
  - Pairs of coordinates: (x1, y1) → (x2, y2)
  - Specify the source piece and the target position (for castling, the king's move is specified)
- **Sensors**:

Example: Chess Agent

- **Performance**:
  - Once reached a terminal state (draw, win): -1 for losing, 0 for draw, 1 for winning
- **Environment**:
  - Chess board, opponent, chess rules
- **Actions**:
  - Pairs of coordinates: (x1, y1) → (x2, y2)
  - Specify the source piece and the target position (for castling, the king's move is specified)
- **Sensors**:
  - Board perceived as a 8x8 matrix. Each element in the matrix can take one of the following values: E, BP, WP, BB, WB, BK, WN, BR, WN, BQ, WK, BK, WK

Example: Chess Agent

- **Performance**:
  - Once reached a terminal state (draw, win): -1 for losing, 0 for draw, 1 for winning
- **Environment**:
  - Chess board, opponent, chess rules
- **Actions**:
  - Pairs of coordinates: (x1, y1) → (x2, y2)
  - Specify the source piece and the target position (for castling, the king's move is specified)
- **Sensors**:
  - Board perceived as a 8x8 matrix. Each element in the matrix can take one of the following values: E, BP, WP, BB, WB, BK, WN, BR, WN, BQ, WK, BK, WK
  - Note that the environment might have lots of features (piece size and material, opponent's facial expression, sunny vs. raining etc.)
  - … but for rational agents we only need to build sensors for the information that is necessary to maximize the performance measure

Example: Chess Agent

Environment

WR	WN	WB	WK	WQ	WB	WN	WR
WP	WP	WP	WP	WP	WP	WP	WP
BP	BP	BP	BP	BP	BP	BP	BP
BR	BN	BB	BK	BQ	BB	BN	BR

Percept

Properties of Environments

- **Observable:** The agent can sense the entire state of the environment.
- **Deterministic**: The outcome of actions is determined by the state of the board and the action executed.
- **Sequential**: Action outcomes can be affected by prior actions (alternative to episodic).
- **Static**: State of the world doesn't change during deliberation (between actions).
- **Discrete**: Actions can only be performed in a small fixed number of ways.
- **Single-Agent**: There is only one agent of change.

Environment Types

	Observable	Deterministic	Sequential	Static	Discrete	Single-Agent
Solitaire						
Chess						
Backgammon						
Taxi						
Angry Birds						
Real Life						

Environment Types

	Observable	Deterministic	Sequential	Static	Discrete	Single-Agent
Solitaire	No	Yes	Yes	Yes	Yes	Yes
Chess						
Backgammon						
Taxi						
Angry Birds						
Real Life						

Environment Types

	Observable	Deterministic	Sequential	Static	Discrete	Single-Agent
Solitaire	No	Yes	Yes	Yes	Yes	Yes
Chess	Yes	Yes	Yes	Yes	Yes	No
Backgammon						
Taxi						
Angry Birds						
Real Life						

Environment Types

	Observable	Deterministic	Sequential	Static	Discrete	Single-Agent
Solitaire	No	Yes	Yes	Yes	Yes	Yes
Chess	Yes	Yes	Yes	Yes	Yes	No
Backgammon	Yes	No	Yes	Yes	Yes	No
Taxi						
Angry Birds						
Real Life						

Environment Types

	Observable	Deterministic	Sequential	Static	Discrete	Single-Agent
Solitaire	No	Yes	Yes	Yes	Yes	Yes
Chess	Yes	Yes	Yes	Yes	Yes	No
Backgammon	Yes	No	Yes	Yes	Yes	No
Taxi	No	No	Yes	No	No	No
Angry Birds						
Real Life						

Environment Types

	Observable	Deterministic	Sequential	Static	Discrete	Single-Agent
Solitaire	No	Yes	Yes	Yes	Yes	Yes
Chess	Yes	Yes	Yes	Yes	Yes	No
Backgammon	Yes	No	Yes	Yes	Yes	No
Taxi	No	No	Yes	No	No	No
Angry Birds	Yes *	Yes	Yes	No	Yes *	Yes
Real Life						

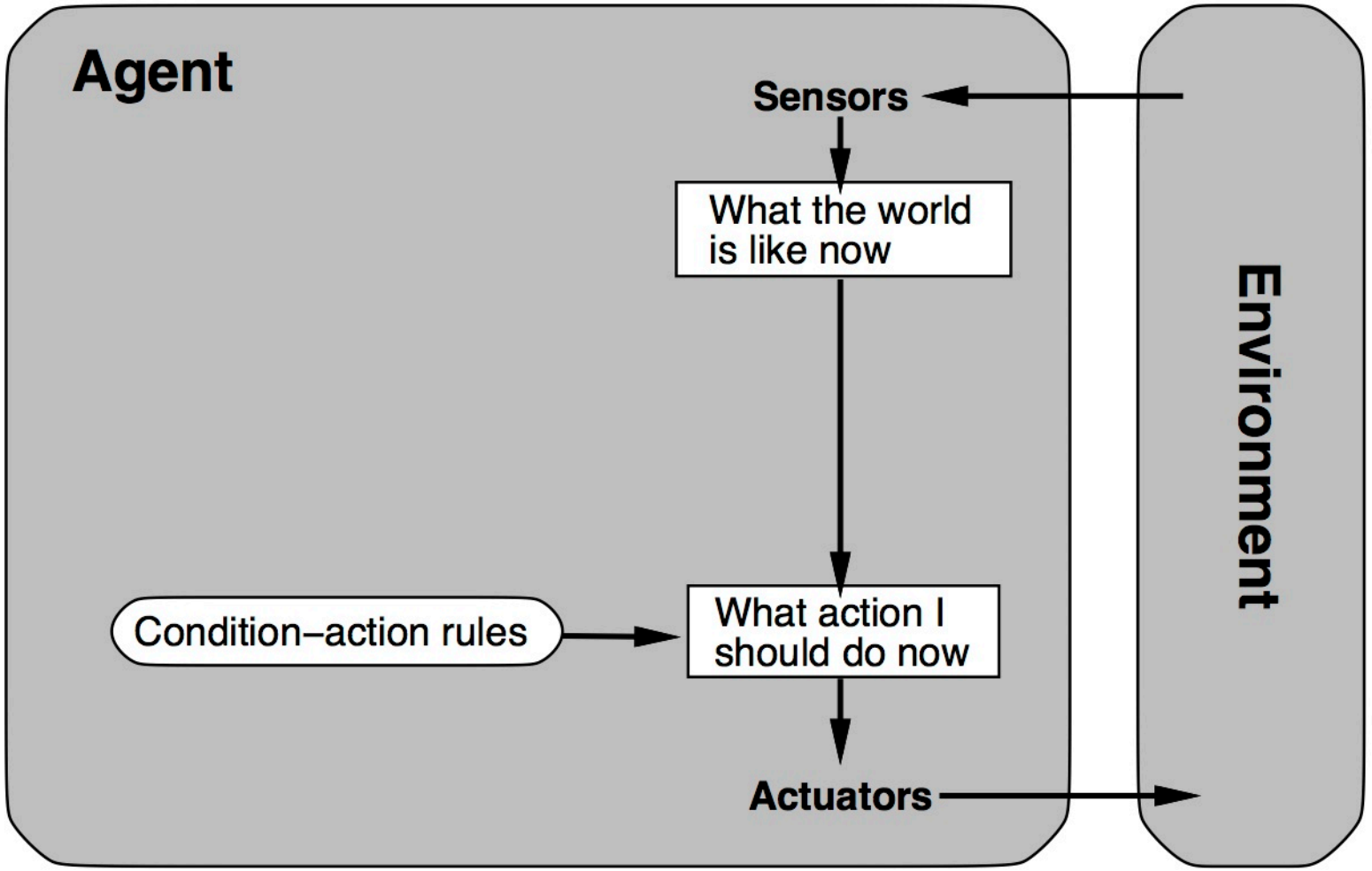
Environment Types

	Observable	Deterministic	Sequential	Static	Discrete	Single-Agent
Solitaire	No	Yes	Yes	Yes	Yes	Yes
Chess	Yes	Yes	Yes	Yes	Yes	No
Backgammon	Yes	No	Yes	Yes	Yes	No
Taxi	No	No	Yes	No	No	No
Angry Birds	Yes *	Yes	Yes	No	Yes *	Yes
Real Life	No	No (?)	Yes	No	No (?)	No

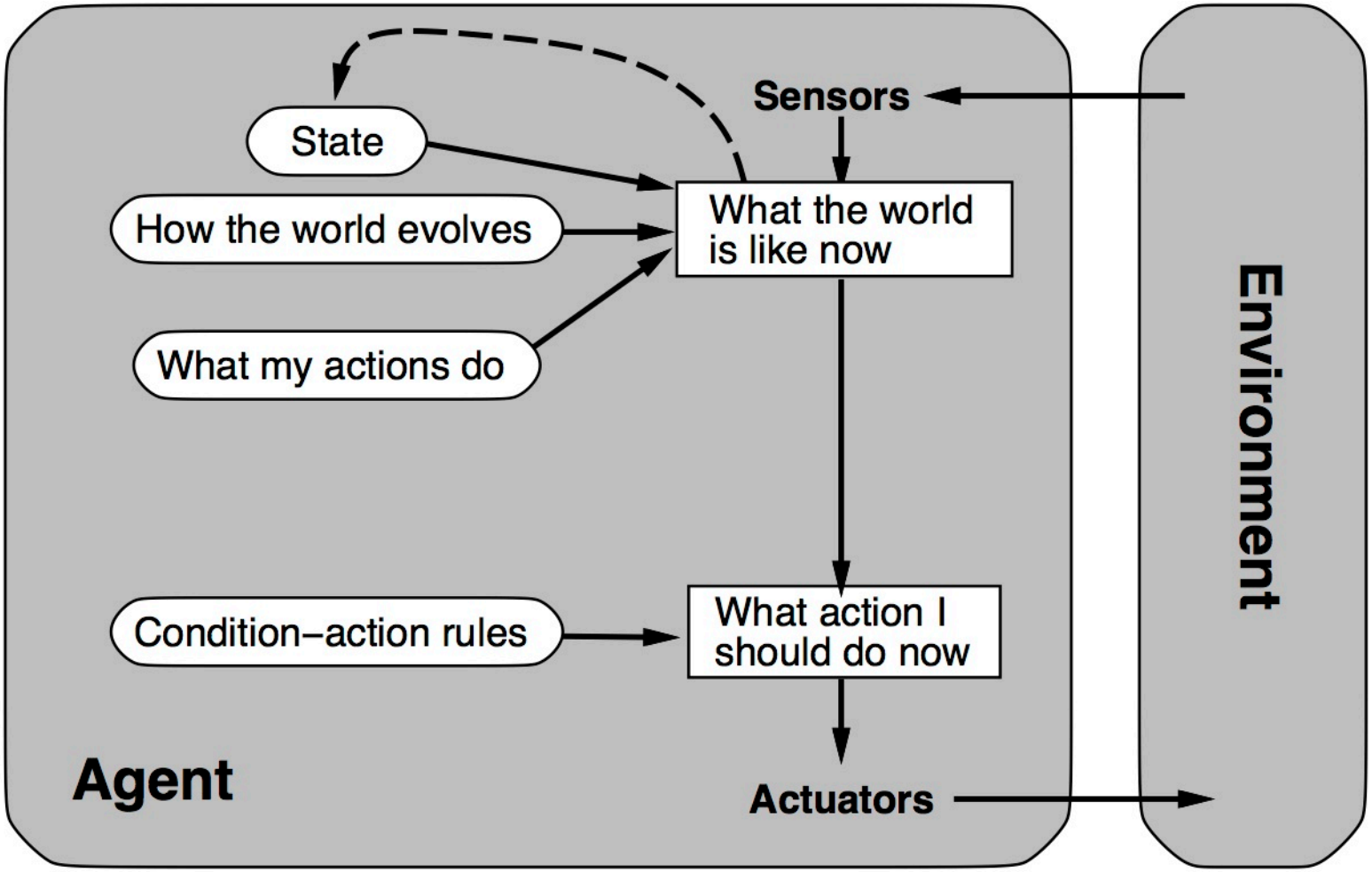
Agent Types

- We will cover 4 basic types:
  - Reactive agents (reflex agents)
  - State-based agents
  - Goal-based agents
  - Utility-based agents
- They can all incorporate learning

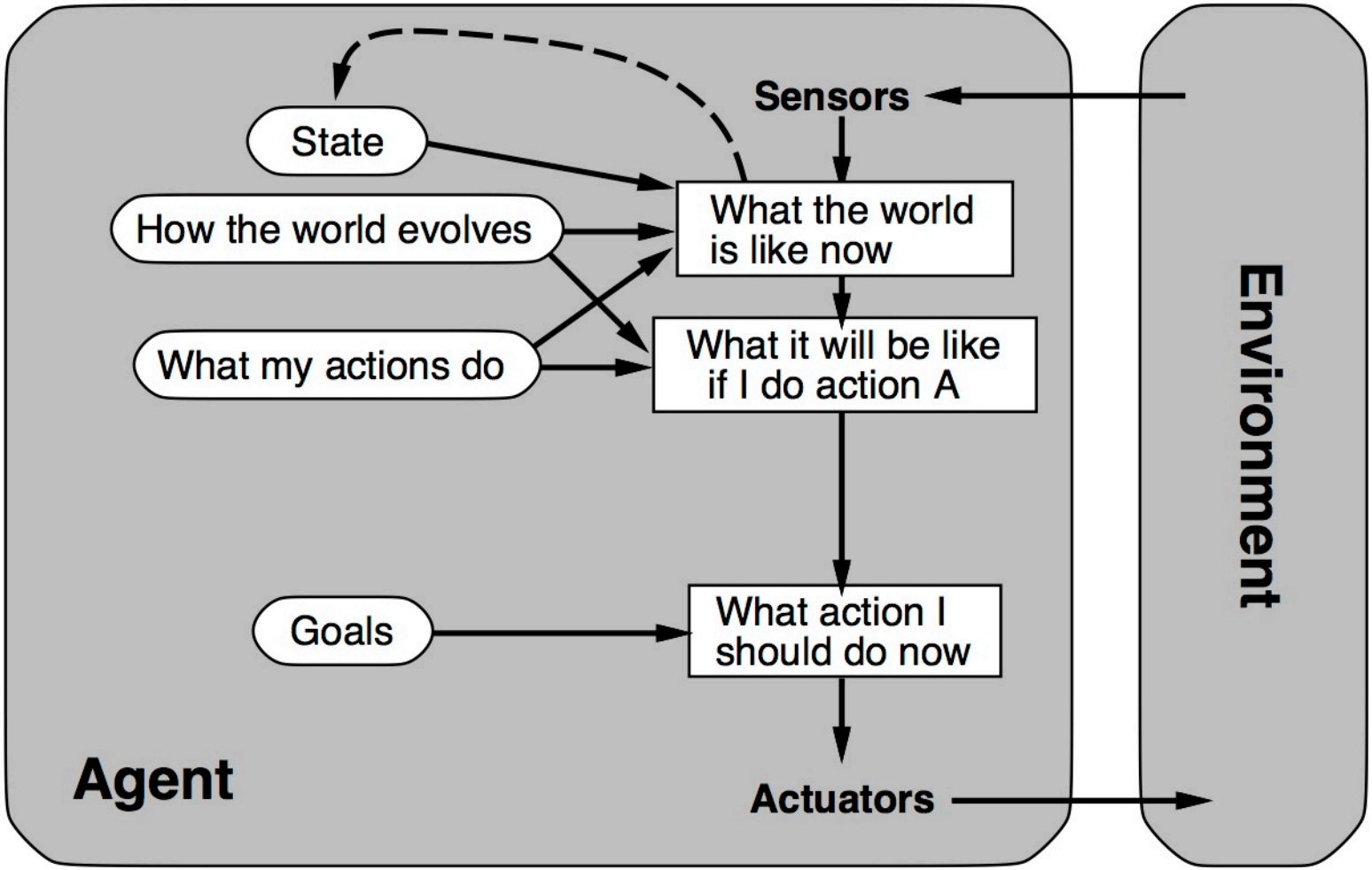
Reactive Agents



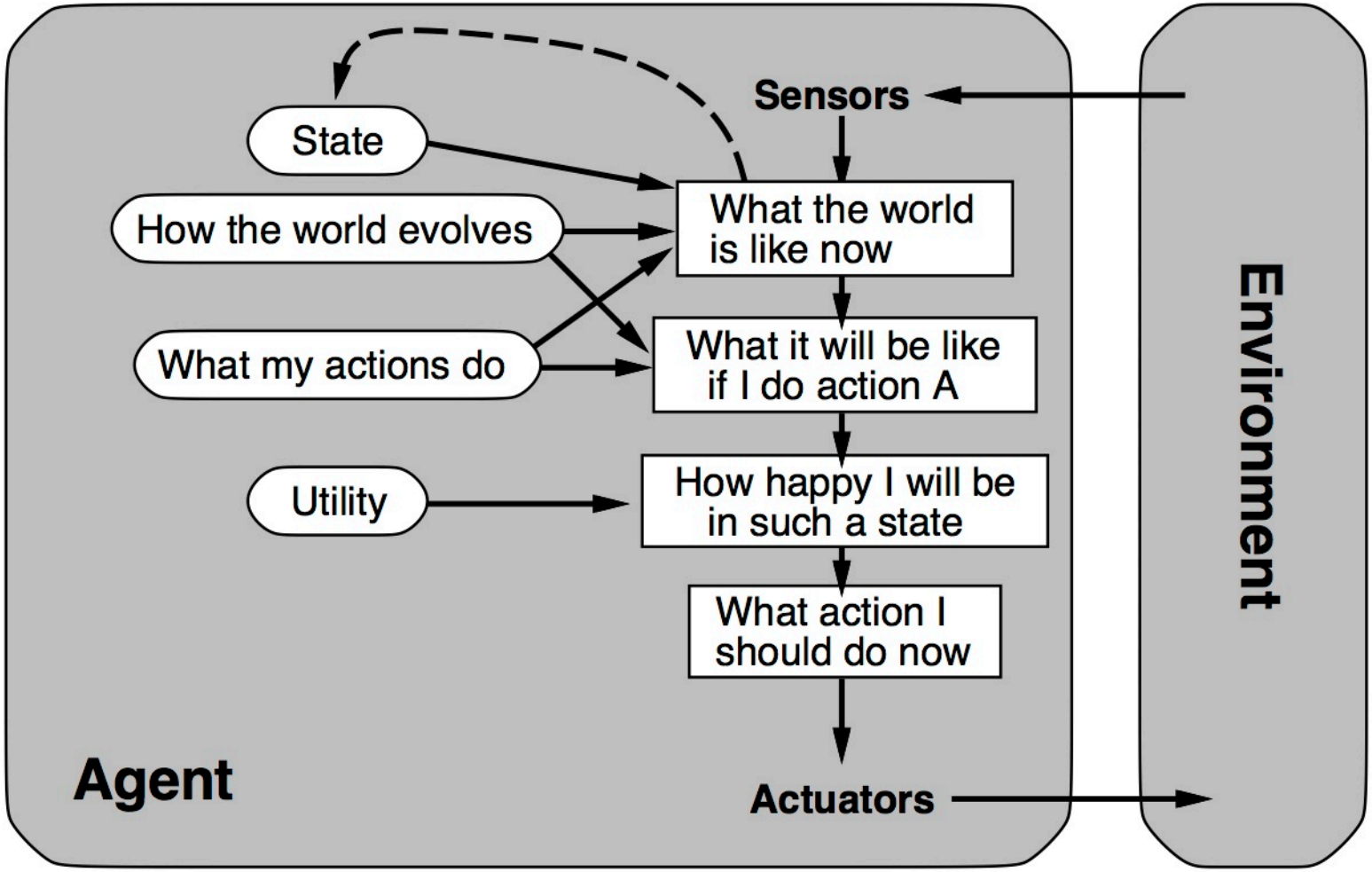
State-based Agents



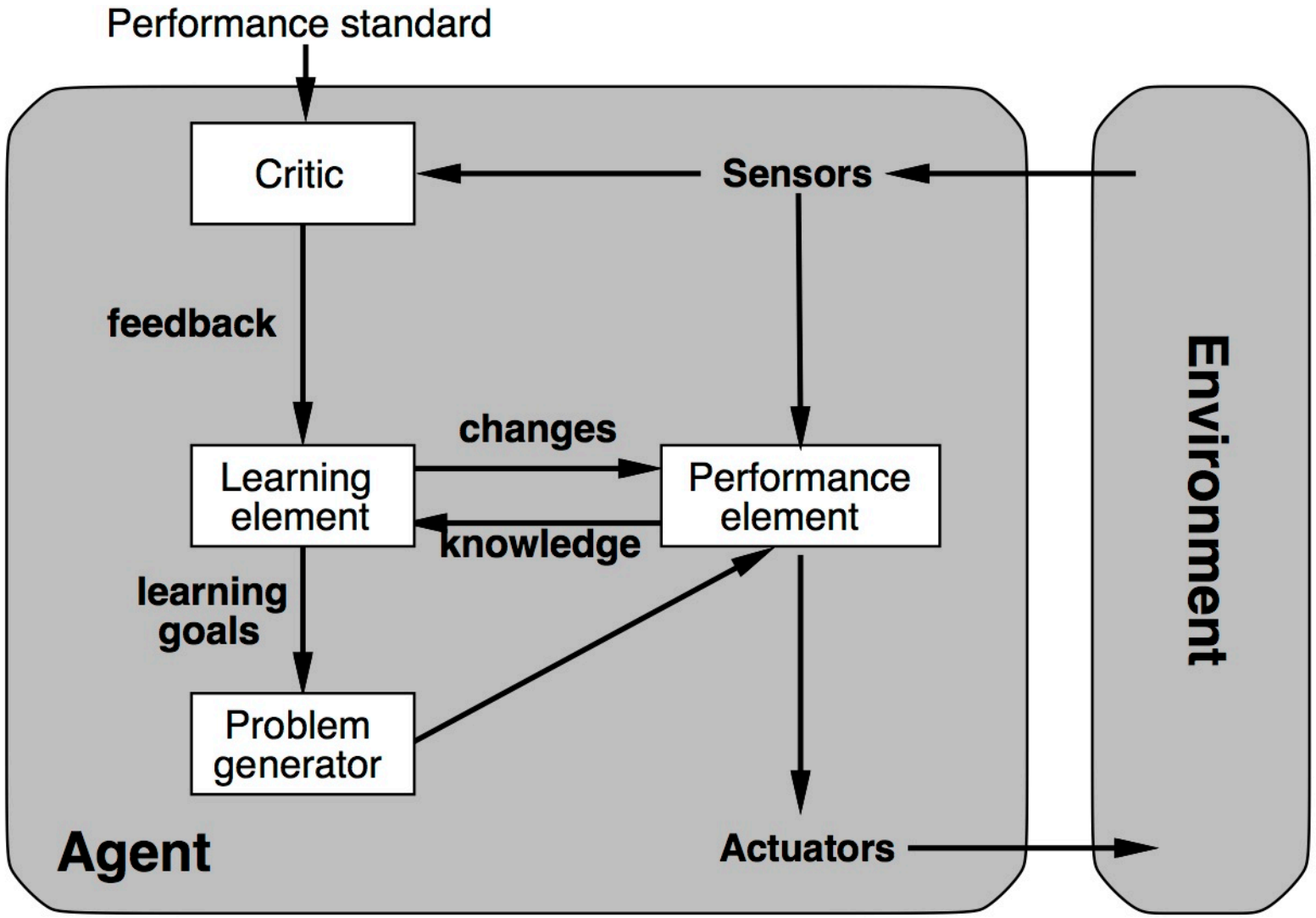
Goal-based Agents



Utility-based Agents



Learning Agents



Looking Ahead…

- What we have covered so far…
  - What is AI?
  - What is a Rational Agent?
- In the next few lectures, we will cover…
  - How can rational agents solve problems?
  - How can rational agents represent knowledge?
  - How can rational agents learn?