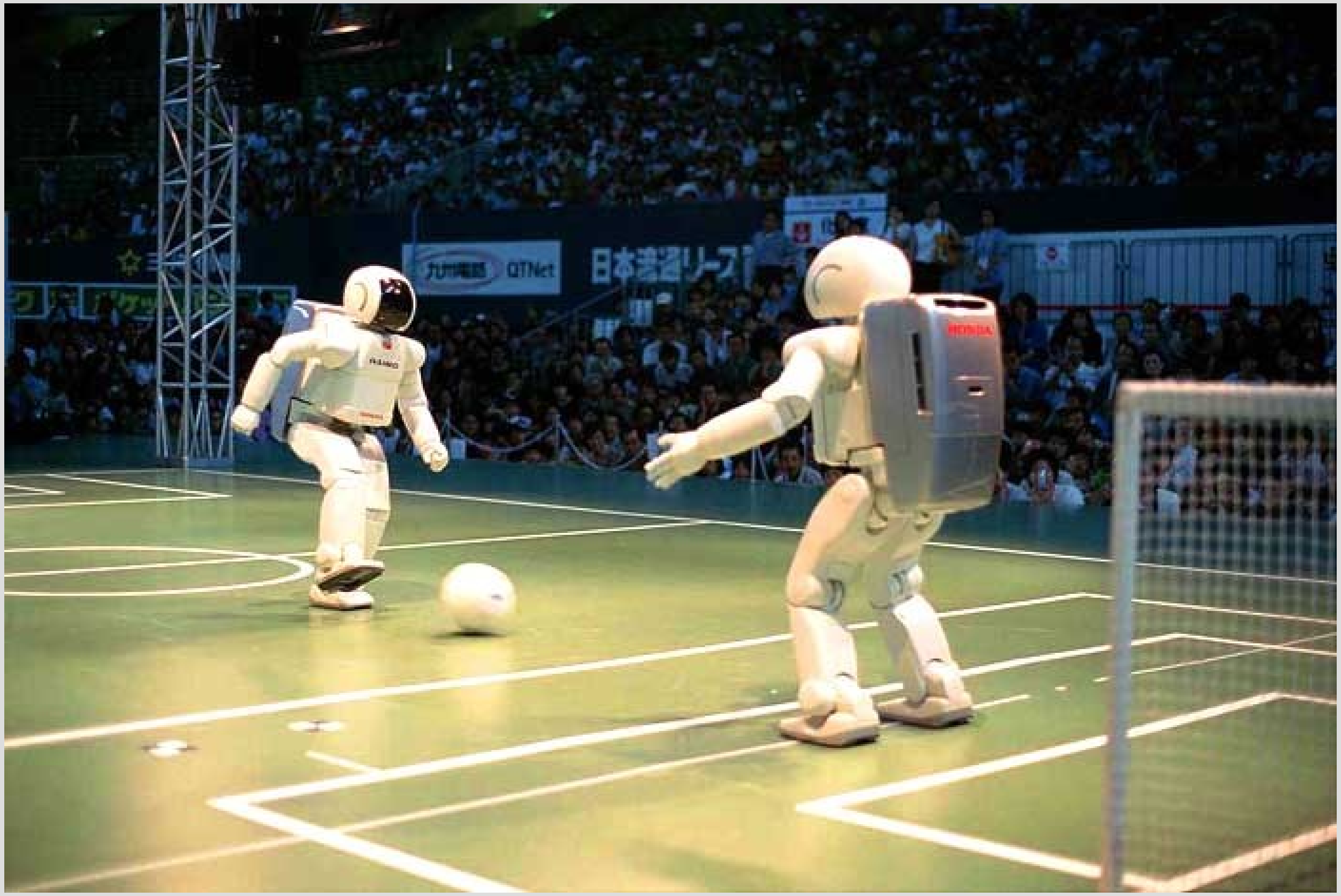
Rational Agents (Ch. 2)

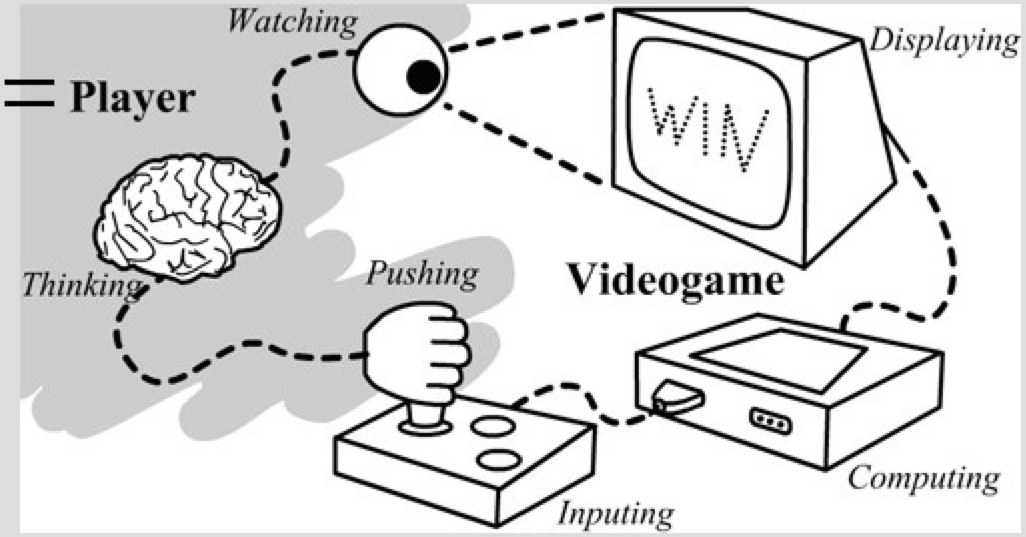


Rational agent

An agent/robot must be able to perceive and interact with the environment

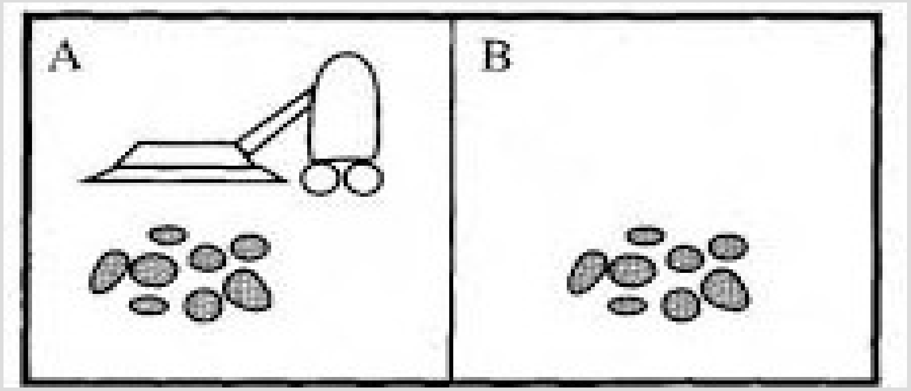
A rational agent is one that always takes the best action (possibly expected best)

Agent =



Rational agent

Consider the case of a simple vacuum agent



Environment: [state A] and [state B], both possibly with dirt that does not respawn

Actions: [move left], [move right] or [suck]

Senses: current location only, [dirty or clean]

Rational agent

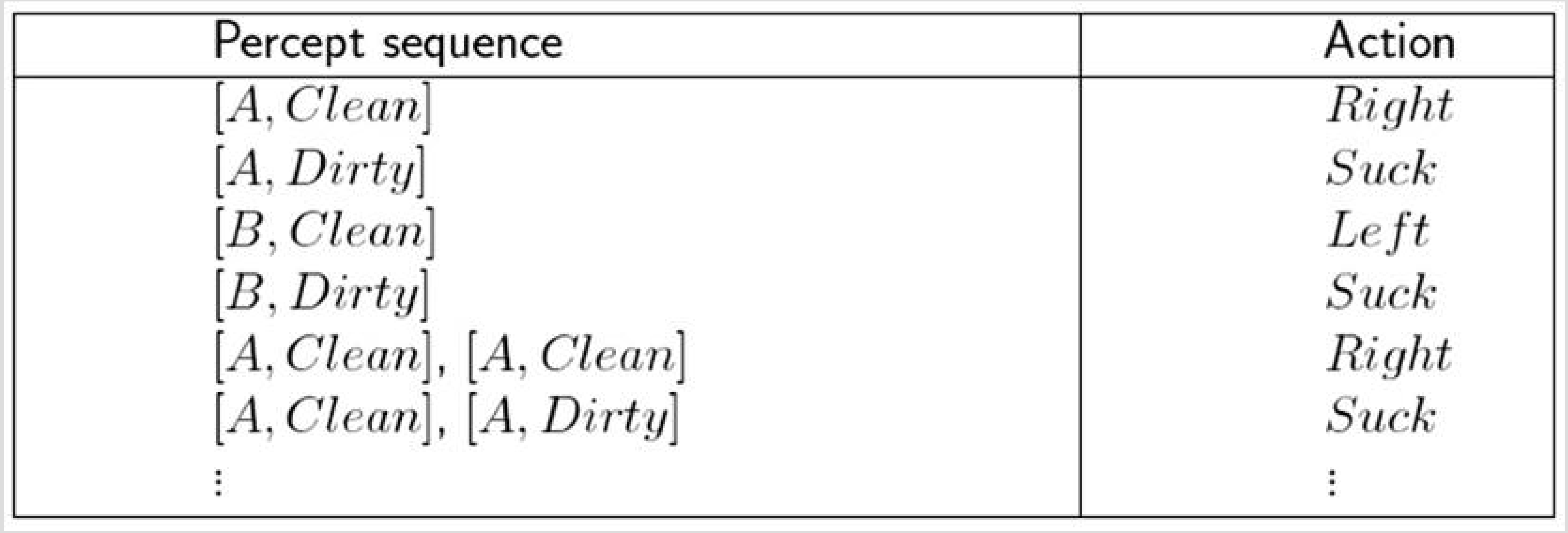
There are two ways to describe an agent's action based on what it sensed:

1. Agent function = directly map what it has seen (and any history) to an action
2. Agent program = logic dictating next action (past and current senses as an input to logic)

The agent function is basically a look-up table, and is typically a much larger code

Rational agent

An agent function for vacuum 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 corresponding agent program:

```
if [Dirty], return [Suck]
if at [state A], return [move right]
if at [state B], return [move left]
```

Rational agent

In order to determine if the vacuum agent is rational I need a performance measure

Under which of these metrics is the agent program on the previous slide rational?

1. Have a clean floor in A and B
2. Have a clean floor as fast as possible
3. Have a clean floor with moving as little as possible
4. Maximize the amount of time sucking

Rational agent

You want to express the performance measure in terms of the environment not the agent

For example, if we describe a measure as:
“Suck up the most dirt”

A rational vacuum agent would suck up dirt then dump it back to be sucked up again...

This will not lead to a clean floor

Rational agent

Performance measure: “-50 points per room dirty after action and -1 point per move”

Is our agent rational (with the proposed agent program) if...

1. Dirt does not reappear
2. Dirt always reappears (after score calculated)
3. Dirt has a 30% chance of reappearing (^^)
4. Dirt reappears but at an unknown rate (^^)

Rational agent

If we do not know how often dirt will reappear, a rational agent might need to learn

Learning can use prior knowledge to estimate how often dirt tends to reappear, but should value actual observations more (its percept)

The agent might need to explore and take sub-optimal short-term actions to find a better long-term solution

Rational agent

To recap, a rational agent depends on:

1. Performance measure
2. Prior knowledge of the environment
3. Actions available
4. History of sensed information

You need to know all of these before you can determine rationality

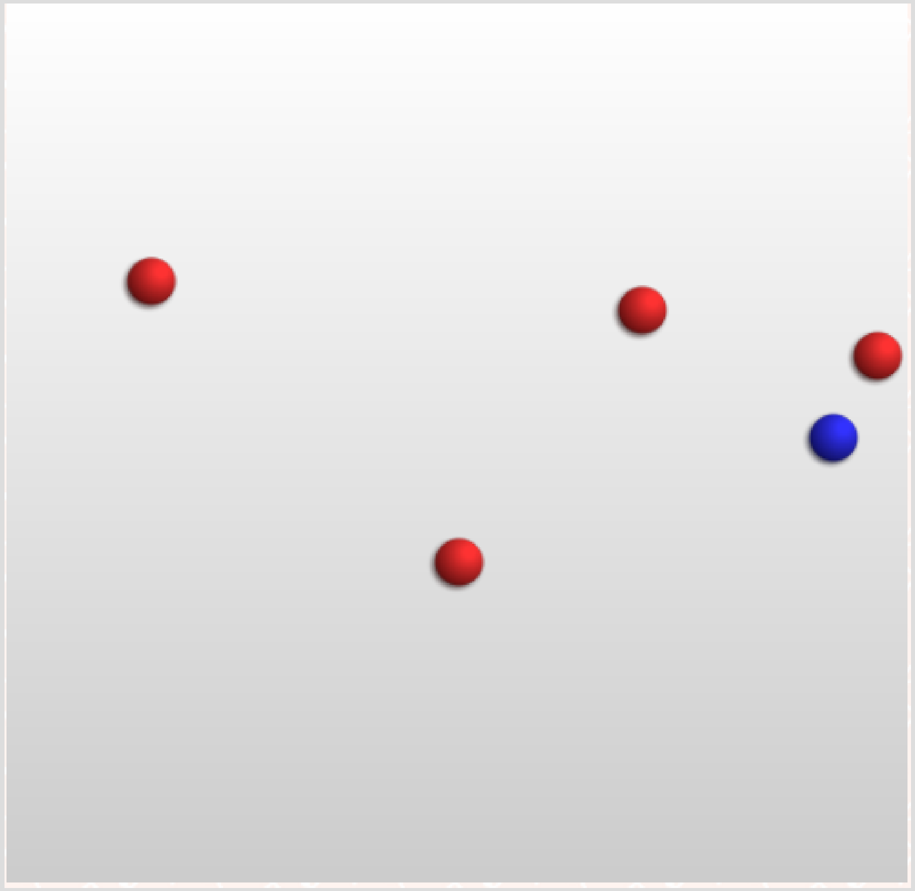
Rational agent

These four items together are called the task environment (abbreviated PEAS)

Performance measure
Environment
Actuators
Sensors

Rational agent

Particle game:
http://www.ragdollsoft.com/particles/



Rational agent

Agent type	Performance	Environment	Actuators	Sensors
Vacuum	time to clean	A, B, dirt	suck, move	dust sensor
Student	GPA, honors	campus, dorm	do HW, take test	eye, ear, hand
Particles	time alive	boarder, red balls	move mouse	screen-shot

Environment classification

Environments can be further classified on the following characteristics:(right side harder)

1. Fully vs. partially observable
2. Single vs. multi-agent
3. Deterministic vs. stochastic
4. Episodic vs. sequential
5. Static vs. dynamic
6. Discrete vs. continuous
7. Known vs. unknown

Environment classification

In a <u>fully observable</u> environment, agents can see every part.

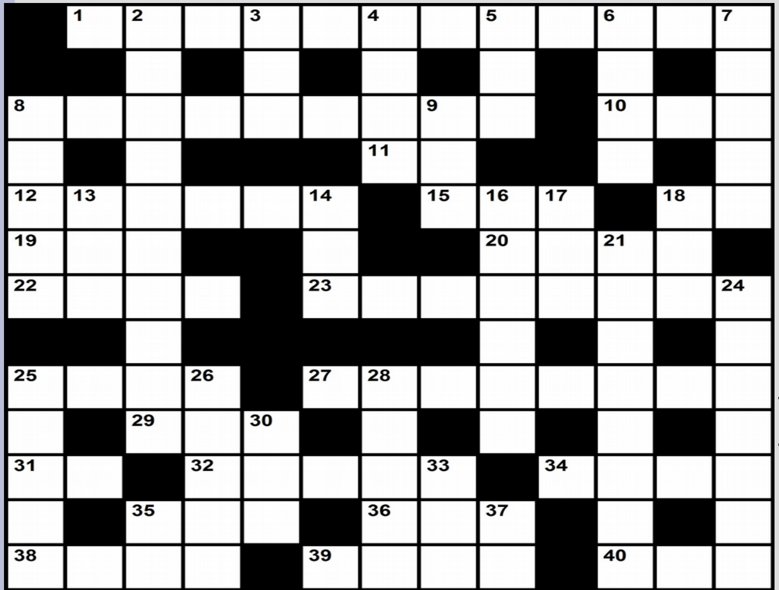
Agents can only see part of the environment if it is <u>partially observable</u>

Full ↗ Partial ⟶

Environment classification

If your agent is the only one, the environment is a single agent environment

More than one is a multi-agent environment (possibly cooperative or competitive)



← single

multi →

Environment classification

If your state+action has a known effect in the environment, it is <u>deterministic</u>

If actions have a distribution (probability) of possible effects, it is <u>stochastic</u>

$\leftarrow$ deterministic

stochastic $\rightarrow$

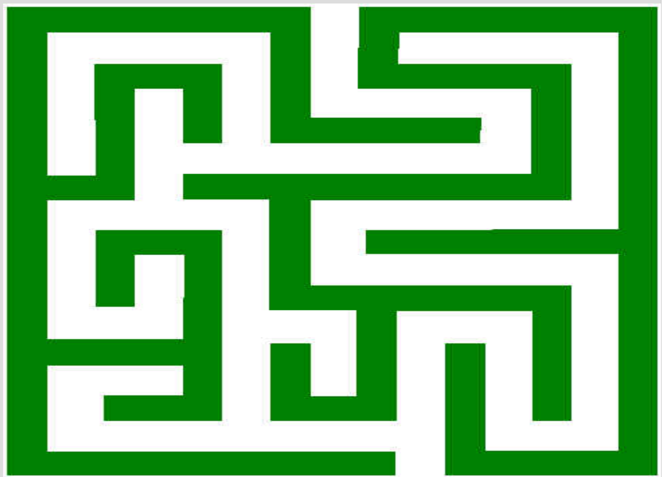
Environment classification

An <u>episodic</u> environment is where the previous action does not effect the next observation (i.e. can be broken into independent events)

If there is the next action depends on the previous, the environment is <u>sequential</u>

$\longleftarrow$ episodic

sequential $\longrightarrow$



Environment classification

If the environment only changes when you make an action, it is <u>static</u>

a <u>dynamic</u> environment can change while your agent is thinking or observing

static

dynamic

Environment classification

Discrete = separate/distinct (events)
Continuous = fluid transition (between events)

This classification can applies: agent's percept and actions, environment's time and states

discrete (state)

continuous (state)

Environment classification

Known = agent's actions have known effects on the environment

Unknown = the actions have an initially unknown effect on the environment (can learn)

know how to stop

do not know how to stop

Environment classification

1. Fully vs. partially observable = how much can you see?
2. Single vs. multi-agent
    = do you need to worry about others interacting?
3. Deterministic vs. stochastic
    = do you know (exactly) the outcomes of actions?
4. Episodic vs. sequential
    = do your past choices effect the future?
5. Static vs. dynamic = do you have time to think?
6. Discrete vs. continuous
    = are you restricted on where you can be?
7. Known vs. unknown
    = do you know the rules of the game?

Environment classification

Some of these classifications are associated with the state, while others with the actions

<u>State:</u> <u>Actions:</u>

1. Fully vs. partially observable
2. Single vs. multi-agent
3. Deterministic vs. stochastic
4. Episodic vs. sequential
5. Static vs. dynamic
6. Discrete vs. continuous
7. Known vs. unknown

Environment classification

Pick a game/hobby/sport/pastime/whatever and describe both the PEAS and whether the environment/agent is:

1. Fully vs. partially observable
2. Single vs. multi-agent
3. Deterministic vs. stochastic
4. Episodic vs. sequential
5. Static vs. dynamic
6. Discrete vs. continuous
7. Known vs. unknown

Environment classification

Agent type	Performance	Environment	Actuators	Sensors
Particles	time alive	boarder, red balls	move mouse	screen-shot

Fully observable, single agent, deterministic, sequential (halfway episodic), dynamic, continuous (time, state, action, and percept), known (to me!)

Agent models

Can also classify agents into four categories:

1. Simple reflex
2. Model-based reflex
3. Goal based
4. Utility based

Top is typically simpler and harder to adapt to similar problems, while bottom is more general representations

Agent models

A simple reflex agents acts only on the most recent part of the percept and not the whole history

Our vacuum agent is of this type, as it only looks at the current state and not any previous

These can be generalized as:
“if state = _____ then do action _____”
(often can fail or loop infinitely)

Agent models

A model-based reflex agent needs to have a representation of the environment in memory (called internal state)

This internal state is updated with each observation and then dictates actions

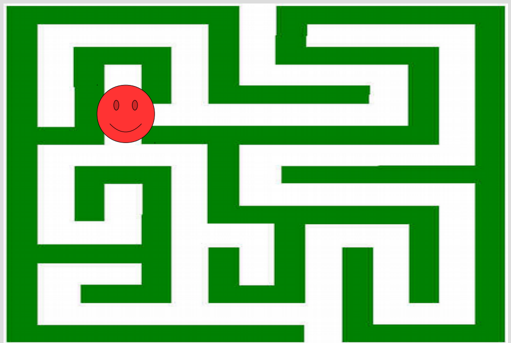
The degree that the environment is modeled is up to the agent/designer (a single bit vs. a full representation)

Agent models

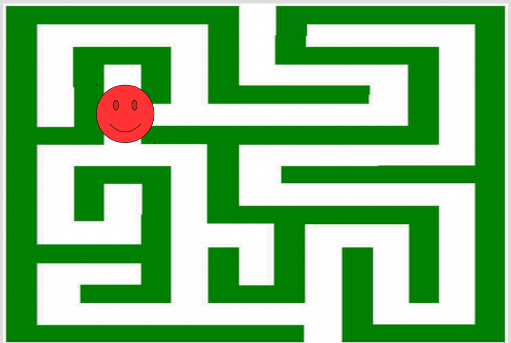
This internal state should be from the agent's perspective, not a global perspective
(as same global state might have different actions)

Consider these pictures of a maze:
Which way to go?

Pic 1



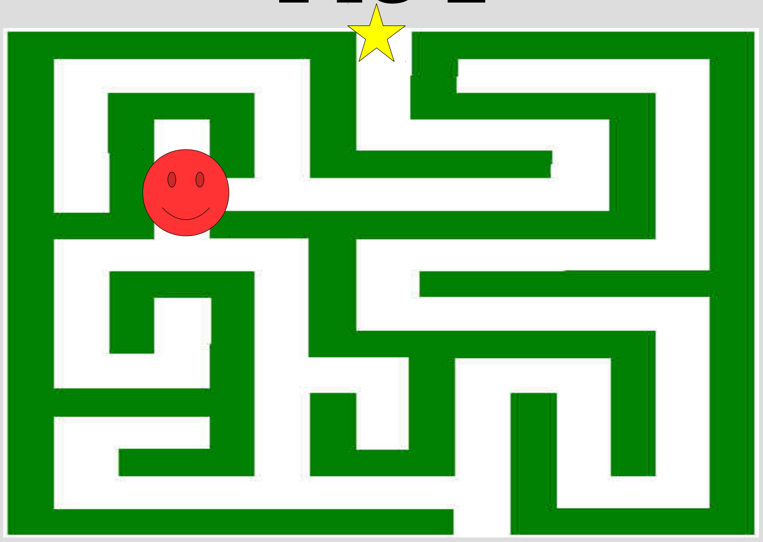
Pic 2



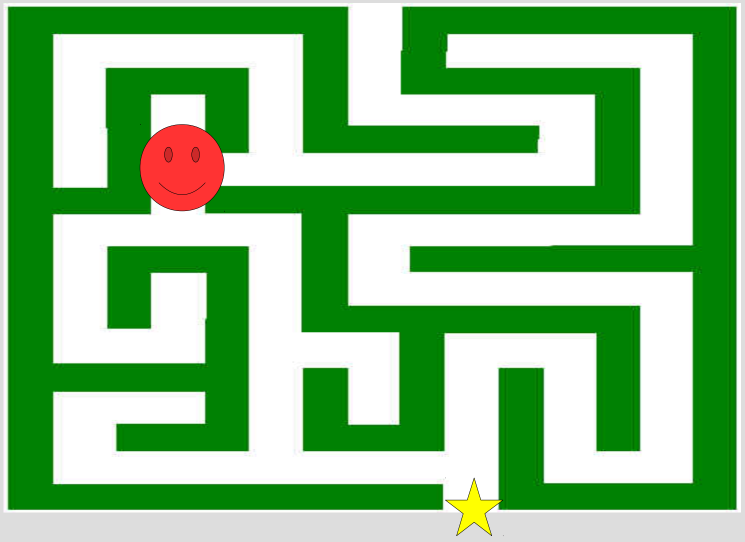
Agent models

The global perspective is the same, but the agents could have different goals (stars)

Pic 1



Pic 2



Goals are not global information

Agent models

For the vacuum agent if the dirt does not reappear, then we do not want to keep moving

The simple reflex agent program cannot do this, so we would have to have some memory (or model)

This could be as simple as a flag indicating whether or not we have checked the other state

Agent models

The goal based agent is more general than the model-based agent

In addition to the environment model, it has a goal indicating a desired configuration

Abstracting to a goals generalizes your method to different (similar) problems
(for example, a model-based agent could solve one maze, but a goal can solve any maze)

Agent models

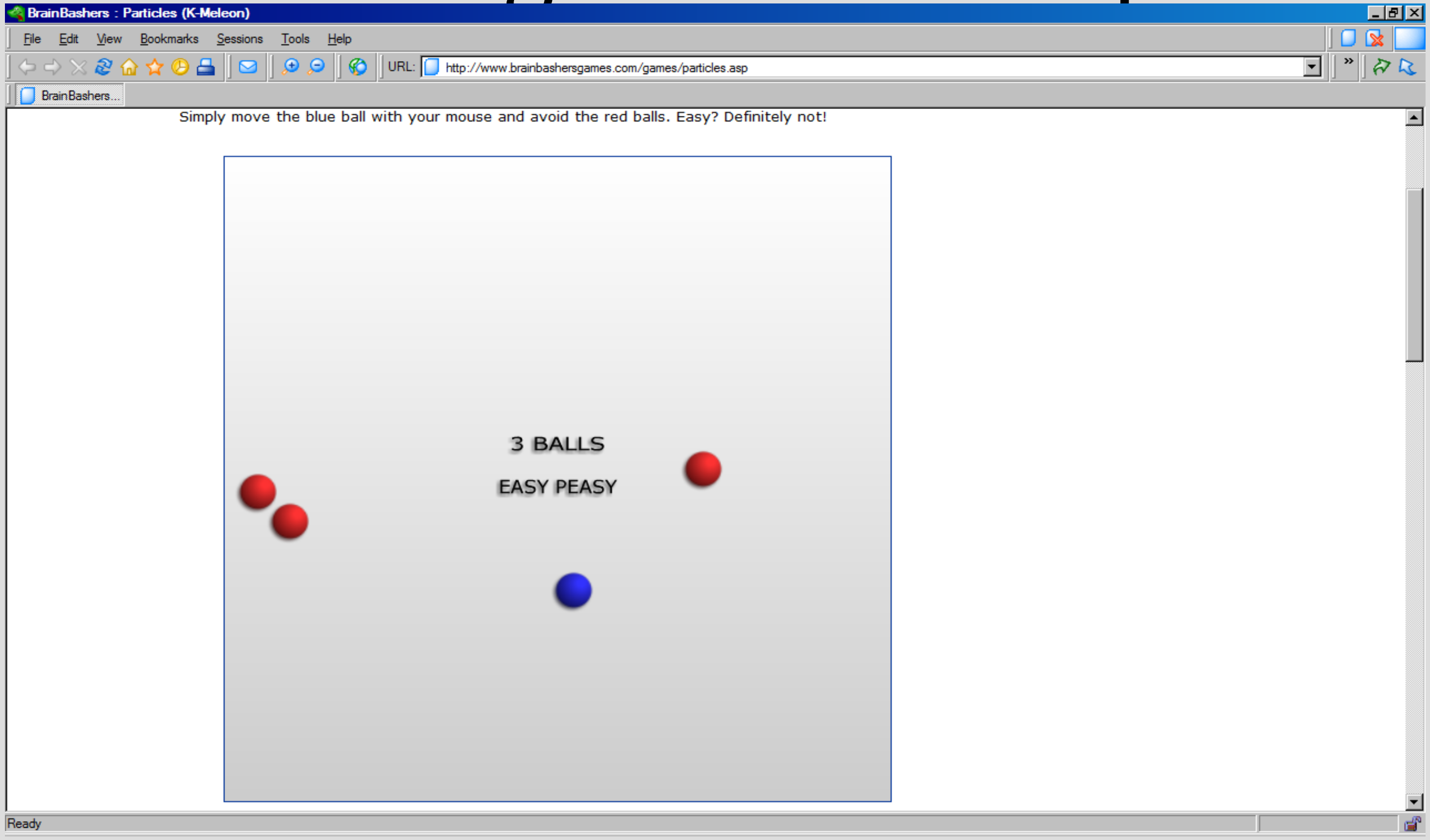
A utility based agent maps the sequence of states (or actions) to a real value

Goals can describe general terms as “success” or “failure”, but there is no degree of success

In the maze example, a goal based agent can find the exit. But a utility based agent can find the shortest path to the exit

Agent models

What is the agent model of particles?


BrainBashers : Particles (K-Meleon)
File Edit View Bookmarks Sessions Tools Help
URL: http://www.brainbashersgames.com/games/particles.asp
BrainBashers...
Ready

Simply move the blue ball with your mouse and avoid the red balls. Easy? Definitely not!

3 BALLS

EASY PEASY

Think of a way to improve the agent and describe what model it is now

State structure

An <u>atomic</u> state has no sub-parts and acts as a simple unique identifier

An example is an elevator:
Elevator = agent (actions = up/down)
Floor = state

In this example, when someone requests the elevator on floor 7, the only information the agent has is what floor it currently is on

State structure

Another example of an atomic representation is simple path finding:
If we start (here) in Vincent 16, how would you get to Keller's CS office?

V. 16 -> Hallway1 -> V. Stairs -> Outside -> Walk to KHKH -> K. Stairs -> CS office

The words above hold no special meaning other than differentiating from each other

State structure

A factored state has a fixed number of variables/attributes associated with it

Our simple vacuum example is factored, as each state has an id (A or B) along with a “dirty” property

In particles, each state has a set of red balls with locations along with the blue ball location

State structure

Structured states simply describe objects and their relationship to others

Suppose we have 3 blocks: A, B and C
We could describe: A on top of B, C next to B

A factored representation would have to enumerate all possible configurations of A, B and C to be as representative

State structure

We will start using <u>structured</u> approaches when we deal with logic:

Summer implies Warm
Warm implies T-Shirt

The current state might be:
!Summer (¬Summer)
but the states have intrinsic relations between each other (not just actions)