Summary/Planning

Artificial Intelligence @ Allegheny College

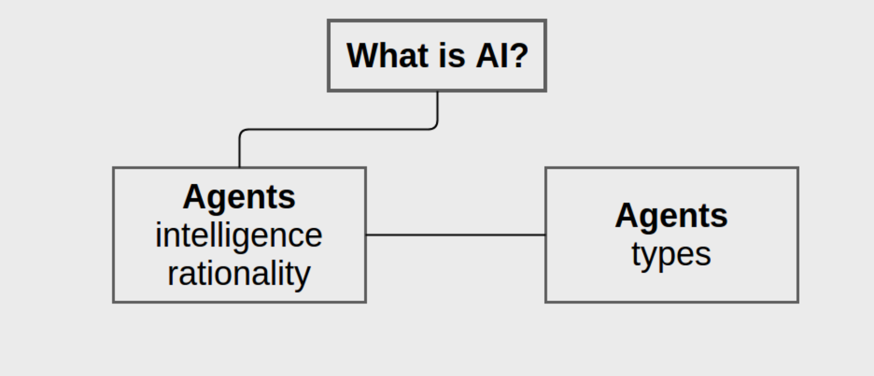
Janyl Jumadinova

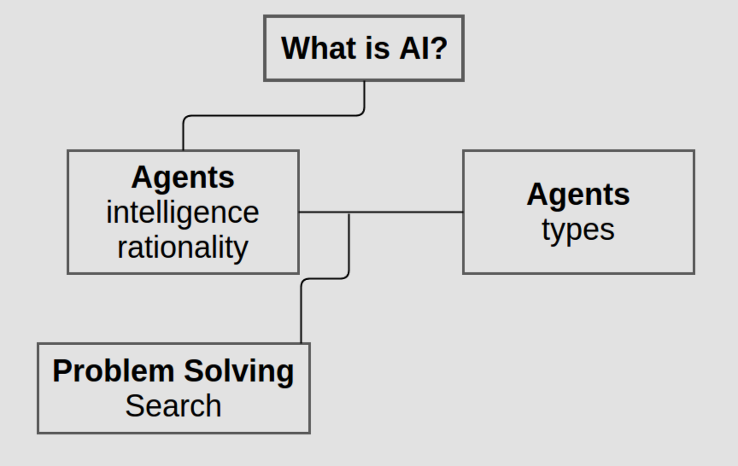
December 1, 2021

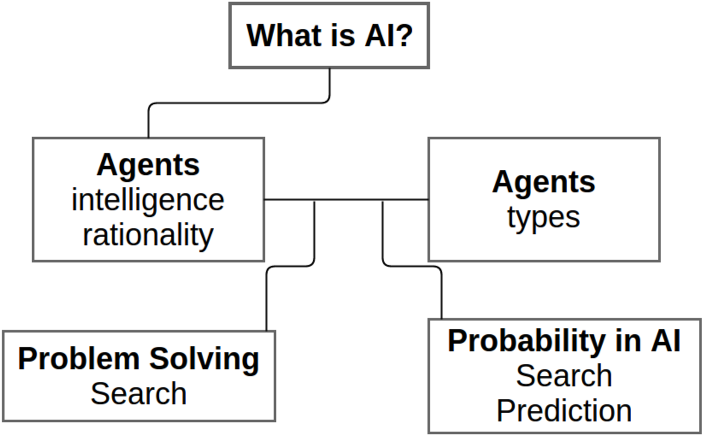
What is AI?

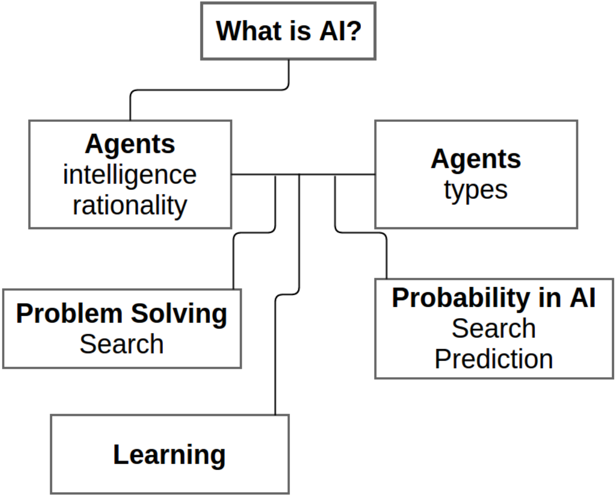
What is AI?

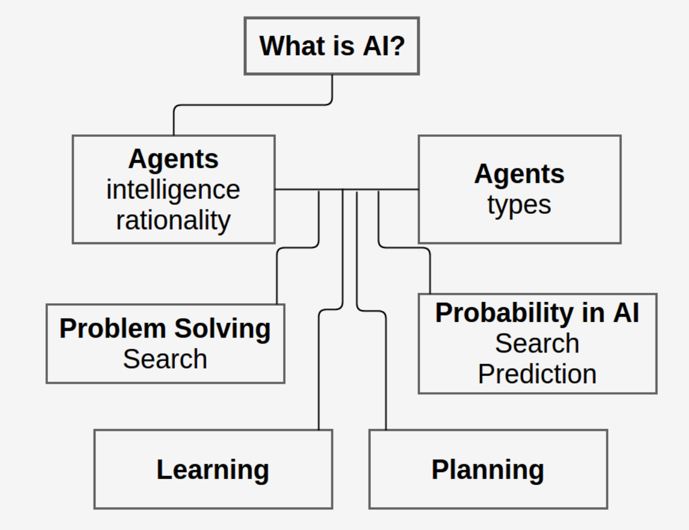
Agents
intelligence
rationality


What is AI?
Agents
intelligence
rationality
Agents
types


What is AI?
Agents
intelligence
rationality
Agents
types
Problem Solving
Search


What is AI?
Agents
intelligence
rationality
Agents
types
Problem Solving
Search
Probability in AI
Search
Prediction


What is AI?
Agents
intelligence
rationality
Agents
types
Problem Solving
Search
Probability in AI
Search
Prediction
Learning


What is AI?
Agents
intelligence
rationality
Agents
types
Problem Solving
Search
Probability in AI
Search
Prediction
Learning
Planning

Planning

Planning:

- deciding what to do based on an agent's ability, its goals, and the state of the world;
- 'thinking head' using logical representations of the states of the world;
- finding a sequence of actions to solve a goal.

Planning

Planning:

- deciding what to do based on an agent's ability, its goals, and the state of the world;
- 'thinking head' using logical representations of the states of the world;
- finding a sequence of actions to solve a goal.

In *classical planning*, the environment is fully observable; deterministic and static with only a single agent.

Planning

- Automation requires efficient automated planning.
- In comparison with the classification problem planning problem solutions provide guarantees on its quality.

Planning

- Automation requires efficient automated planning.
- In comparison with the classification problem planning problem solutions provide guarantees on its quality.

When to use Planning:

- Desire a procedural course of action for a declaratively described system.
- Domain knowledge can be elicited or learned over time.
- Consistency is more desired than learning transient behaviors.

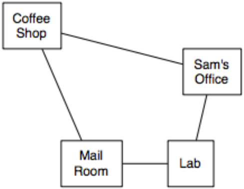
Applications

- System control (autonomous/virtual agents)
- Process control:
  - Construction and configuration
  - Workflow management
  - Mission planning
  - Project planning

Basic planning problem:

- **Given**: start state, goal conditions, actions
- **Find**: sequence of actions leading from start to goal
- **Typically**: states correspond to possible worlds; actions and goals specified using a logical formalism (e.g., STRIPS, situation calculus, temporal logic, etc.)

Delivery Robot Example



Features:

RLoc – Rob's location
RHC – Rob has coffee
SWC – Sam wants coffee
MW – Mail is waiting
RHM – Rob has mail

Actions:

mc – move clockwise
mcc – move counterclockwise
puc – pickup coffee
dc – deliver coffee
pum – pickup mail
dm – deliver mail

Explicit State-space Representation

Enumerate the states and, for each state, specify the actions that are possible in that state and, for each state-action pair, specify the state that results from carrying out the action in that state.

Explicit State-space Representation

Enumerate the states and, for each state, specify the actions that are possible in that state and, for each state-action pair, specify the state that results from carrying out the action in that state.

$4x2x2x2x2 = 64$ states

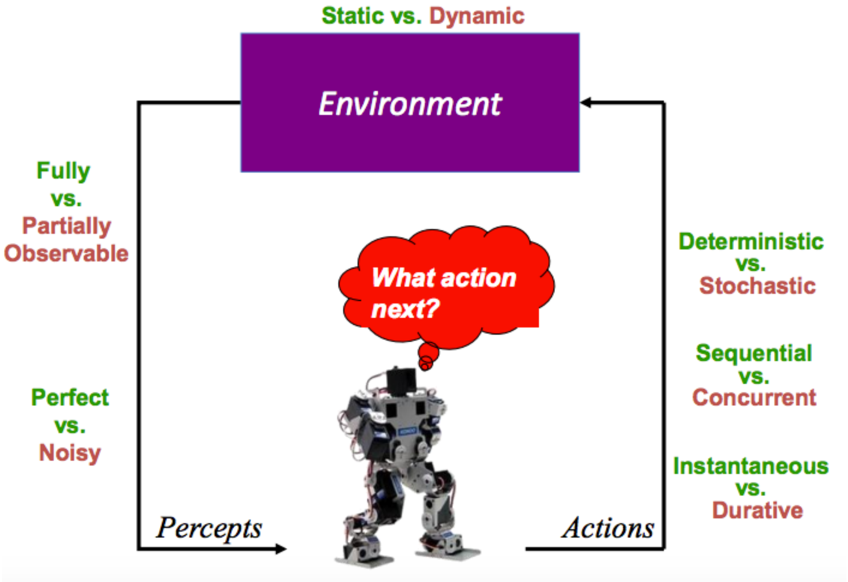
State	Action	Resulting State
$\langle lab, \overline{rhc}, swc, \overline{mw}, rhm \rangle$	mc	$\langle mr, \overline{rhc}, swc, \overline{mw}, rhm \rangle$
$\langle lab, \overline{rhc}, swc, \overline{mw}, rhm \rangle$	mcc	$\langle off, \overline{rhc}, swc, \overline{mw}, rhm \rangle$
$\langle off, \overline{rhc}, swc, \overline{mw}, rhm \rangle$	dm	$\langle off, \overline{rhc}, \overline{swc}, \overline{mw}, \overline{rhm} \rangle$
$\langle off, \overline{rhc}, swc, \overline{mw}, rhm \rangle$	mcc	$\langle cs, \overline{rhc}, swc, \overline{mw}, rhm \rangle$
$\langle off, \overline{rhc}, swc, \overline{mw}, rhm \rangle$	mc	$\langle lab, \overline{rhc}, swc, \overline{mw}, rhm \rangle$
...	...	...

Status of Classical Planning

- **Classical planning works!**
  - Large problems solved very fast (non-optimally)

Status of Classical Planning

- **Classical planning works!**
  - Large problems solved very fast (non-optimally)
- **Limitations:**
  - Does not model **uncertainty** (no probabilities)
  - Does not deal with **incomplete information** (no sensing)
  - Deals with very **simple cost structure** (no state dependent costs)



Static vs. Dynamic
Environment
Fully vs. Partially Observable
Perfect vs. Noisy
What action next?
Deterministic vs. Stochastic
Sequential vs. Concurrent
Instantaneous vs. Durative
Percepts
Actions

Policies

A stationary policy is a function:

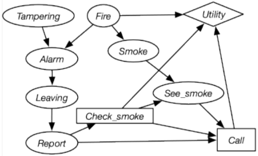
$$\pi : S \to A$$

Given a state s, $\pi(s)$ specifies what action the agent who is following π will do.

An optimal policy is one with maximum expected discounted reward.

Policy Examples

1. Never check for smoke, and call only if there is a report.
2. Always check for smoke, and call only if it sees smoke.
3. Check for smoke if there is a report, and call only if there is a report and it sees smoke.
4. Check for smoke if there is no report, and call when it does not see smoke.
5. Always check for smoke and never call.



Markov Decision Processes (MDPs)

A fundamental framework for probabilistic planning.
MDPs are fully observable, probabilistic state models:

- a state space S
- a set $G \subset S$ of goal states
- actions $A(s) \subset A$ applicable in each state $s \in S$
- transition probabilities $P_a(s_0|s)$ for $s \in S$ and $a \in A(s)$
- action costs $c(a, s) > 0$

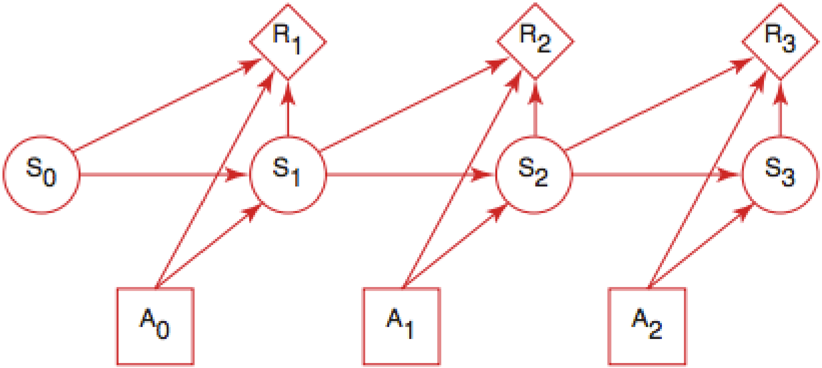
Markov Decision Processes (MDPs)

A fundamental framework for probabilistic planning.
MDPs are fully observable, probabilistic state models:

- a state space S
- a set $G \subset S$ of goal states
- actions $A(s) \subset A$ applicable in each state $s \in S$
- transition probabilities $P_a(s_0|s)$ for $s \in S$ and $a \in A(s)$
- action costs $c(a, s) > 0$

Solutions are functions (**policies**) mapping states into actions.
Optimal solutions have minimum **expected** costs.

Markov Decision Processes (MDPs)



MDP Solution

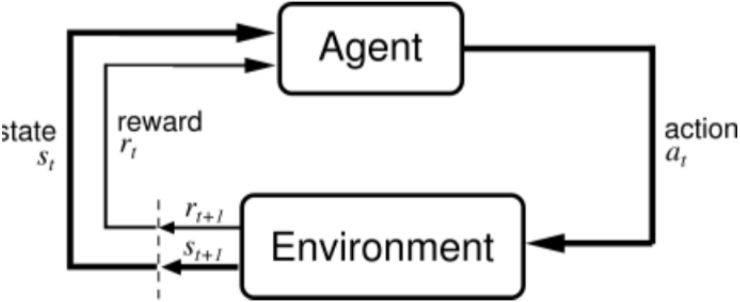
- Want a way to choose an action in a state, i.e., a policy π
- What does a policy look like?
  - Can pick action based on states visited + actions used so far, i.e., execution history $h = s(1)a(1)s(2)a(2)...$
  - Can pick actions randomly

MDP Solution

- Want a way to choose an action in a state, i.e., a policy π
- What does a policy look like?
  - Can pick action based on states visited + actions used so far, i.e., execution history $h = s(1)a(1)s(2)a(2)...$
  - Can pick actions randomly
- Thus, in general an MDP solution is a probabilistic history-dependent $\pi : HxA \rightarrow [0, 1]$

Evaluating MDP Solutions

- Executing a policy yields a sequence of rewards



Evaluating MDP Solutions

- Define utility function $u(R1, R2, ...)$ to be some "quality measure" of a reward sequence.
- Define **value** function as $V : H \rightarrow [-infty, infty]$.
- Define value function of a policy after history h to be some utility function of subsequent rewards:
  $V^{\pi}(h) = u(R1, R2, ...)$

Optimal MDP Solution

- **Want**: a behavior that is "best" in every situation
- π^* is an optimal policy if $V^*(h) \geq V^\pi(h)$ for all π, for all h

Example: to exercise or not?

Example: to exercise or not?

Each week *Sam* has to decide whether to exercise or not:
States: $\{$fit, unfit$\}$ **Actions**: $\{$exercise, relax$\}$

Example: to exercise or not?

Each week *Sam* has to decide whether to exercise or not:
States: {fit, unfit} **Actions**: {exercise, relax}

Dynamics:

State	Action	$P(fit \mid State, Action)$
fit	exercise	0.99
fit	relax	0.7
unfit	exercise	0.2
unfit	relax	0.0

Reward (does not depend on resulting state):

State	Action	Reward
fit	exercise	8
fit	relax	10
unfit	exercise	0
unfit	relax	5

Information Availability

What information is available when the agent decides what to do?

- **Fully-observable MDP** the agent gets to observe S_t when deciding on action A_t.

Information Availability

What information is available when the agent decides what to do?

- **Fully-observable MDP** the agent gets to observe S_t when deciding on action A_t.
- **Partially-observable MDP (POMDP)** the agent has some noisy sensor of the state. It needs to remember its sensing and acting history.

AI Planning at IBM