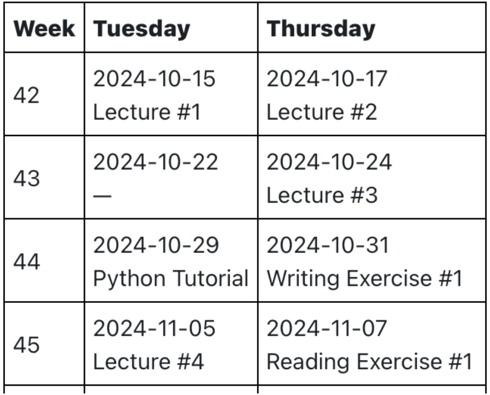
Computational Intelligence

LMU Munich
winter term 2024/2025

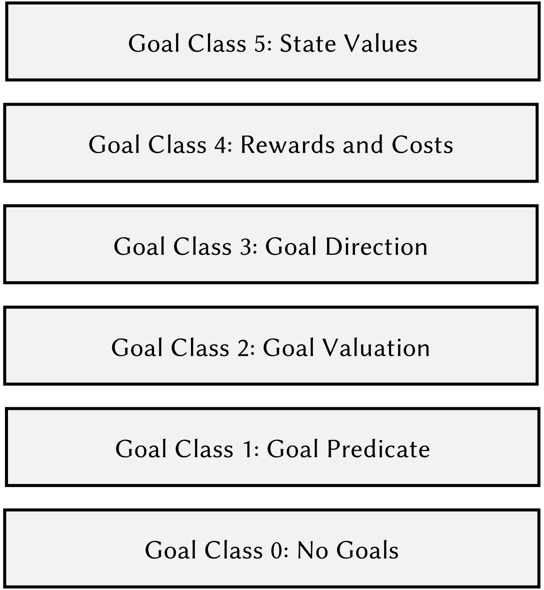
Thomas Gabor
Claudia Linnhoff-Popien

calendar overview



Week	Tuesday	Thursday
42	2024-10-15 Lecture #1	2024-10-17 Lecture #2
43	2024-10-22 —	2024-10-24 Lecture #3
44	2024-10-29 Python Tutorial	2024-10-31 Writing Exercise #1
45	2024-11-05 Lecture #4	2024-11-07 Reading Exercise #1

The Goal Class Hierarchy



Goal Class 5: State Values

Goal Class 4: Rewards and Costs

Goal Class 3: Goal Direction

Goal Class 2: Goal Valuation

Goal Class 1: Goal Predicate

Goal Class 0: No Goals

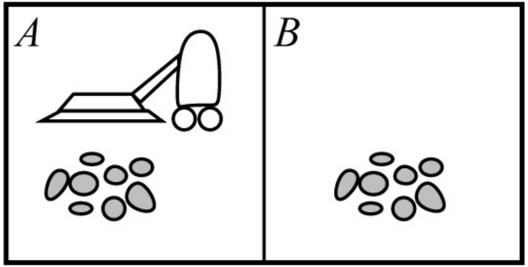
recap

Goal Class 1: Goal Predicate

“I know it is good when I see it!”

running example #1

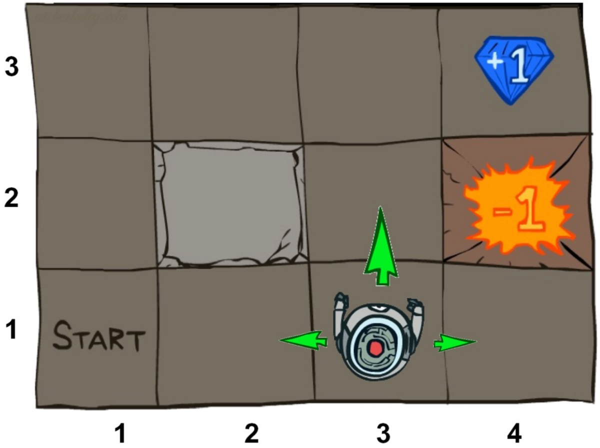
The Vacuum World



Russel, Norvig. Artificial Intelligence – A Modern Approach. Third Edition. 2016.

running example #2

The Basic Grid World



https://inst.eecs.berkeley.edu/~cs188/fa22/

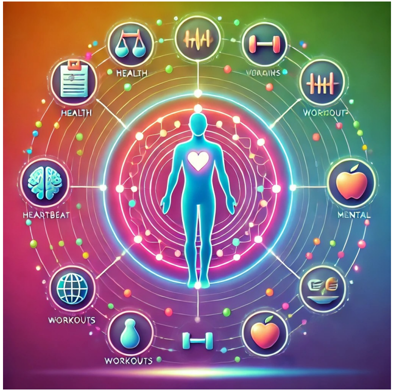
running example #3

Resource/Stock Trading

chatgpt.com

running example #4

Personal Life Assistant



chatgpt.com

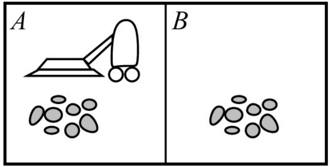
encoding policies…

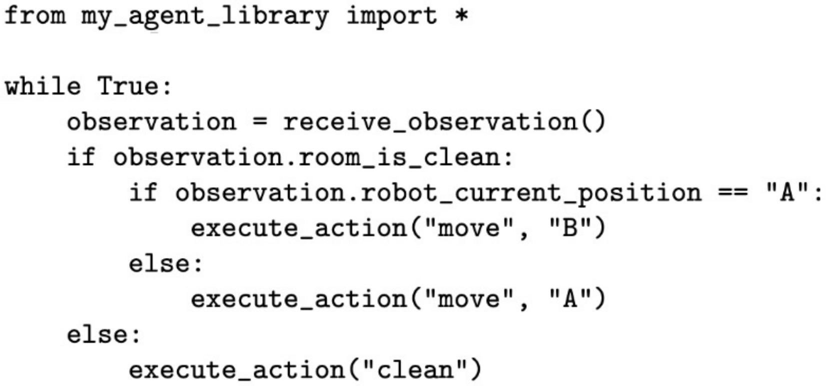
encoding policies...

Arbitrary Functions

encoding policies...

Code in a Programming Language





```
from my_agent_library import *

while True:
    observation = receive_observation()
    if observation.room_is_clean:
        if observation.robot_current_position == "A":
            execute_action("move", "B")
        else:
            execute_action("move", "A")
    else:
        execute_action("clean")
```

Russel, Norvig. Artificial Intelligence – A Modern Approach. Third Edition. 2016.

encoding policies...

Process Notation

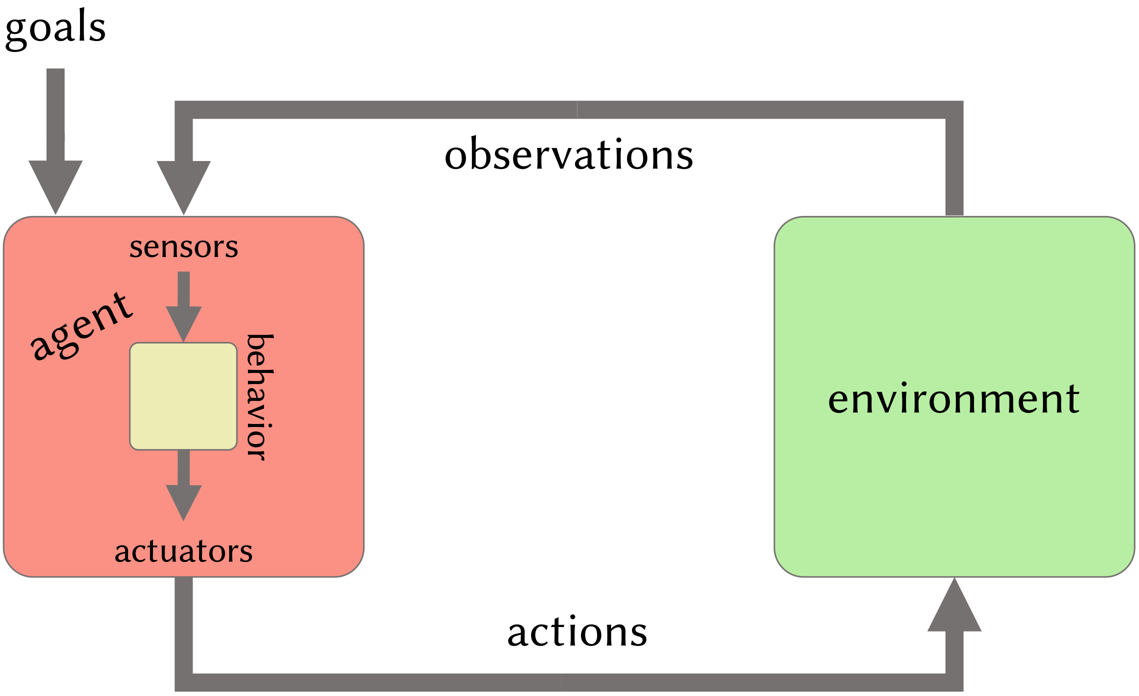
$$\mathit{behavior\ for\ Robot} \begin{cases} \mathit{is\_clean}(\mathit{True}).\mathit{position}(\text{“A”}).\overline{\mathit{move}}\langle\text{“B”}\rangle.\mathit{Robot} \\ \mathit{is\_clean}(\mathit{True}).\mathit{position}(\text{“B”}).\overline{\mathit{move}}\langle\text{“A”}\rangle.\mathit{Robot} \\ \mathit{is\_clean}(\mathit{False}).\mathit{position}(\text{“A”}).\overline{\mathit{clean}}.\mathit{Robot} \\ \mathit{is\_clean}(\mathit{False}).\mathit{position}(\text{“B”}).\overline{\mathit{clean}}.\mathit{Robot} \end{cases}$$

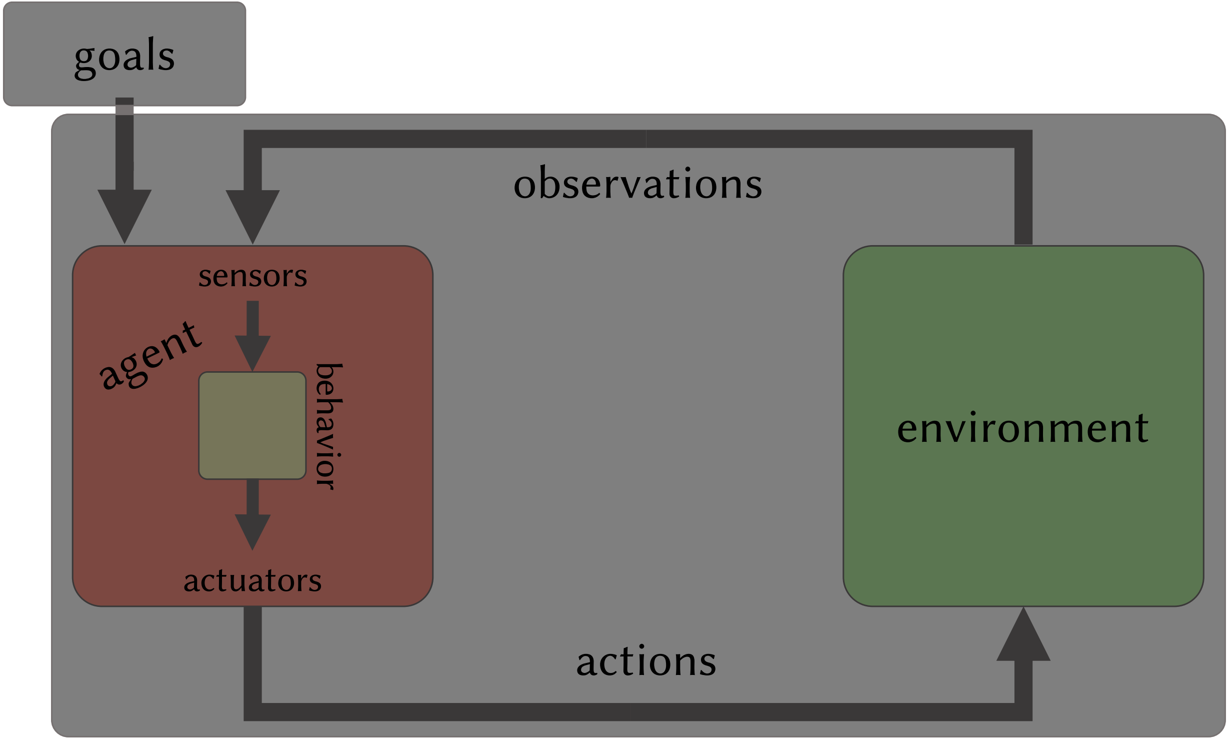
encoding policies...

Process Notation

$$\textit{behavior for Robot}\ \left\{\begin{array}{l} is\_clean(True).position(\text{``A''}).\overline{move}\langle\text{``B''}\rangle.Robot \\ is\_clean(True).position(\text{``B''}).\overline{move}\langle\text{``A''}\rangle.Robot \\ is\_clean(False).position(\text{``A''}).\overline{clean}.Robot \\ is\_clean(False).position(\text{``B''}).\overline{clean}.Robot \end{array}\right.$$

$$\begin{aligned} Robot = is\_clean(True).(&position(\text{``A''}).\overline{move}\langle\text{``B''}\rangle.Robot \\ &+ position(\text{``B''}).\overline{move}\langle\text{``A''}\rangle.Robot) \\ + is\_clean(False).&position(\_).\overline{clean}.Robot \end{aligned}$$


goals
observations
sensors
agent
behavior
actuators
environment
actions


goals
observations
sensors
agent
behavior
actuators
environment
actions

finding policies…

finding policies...

Algorithm 1 (brute force (policy)). Let $\mathcal{A}$ be a set of actions. Let $\mathcal{O}$ be a set of observations. Let $\Gamma \subseteq (\mathcal{O} \to \mathcal{A}) \to \mathbb{B}$ be a space of goal predicates on policy functions. Let $\gamma \in \Gamma$ be a goal predicate. We assume that the policy space $\Pi \subseteq \mathcal{O} \to \mathcal{A}$ is enumerable, i.e., $\Pi = \langle \pi_i \rangle_{i \in \mathbb{N}}$. Brute force starting from i is given via the function

$$b(i) = \begin{cases} \pi_i & \text{if } \gamma(\pi_i), \\ b(i+1) & \text{otherwise.} \end{cases}$$

If not further specified, the call to $b(0)$ is called brute force search for an agent policy. Usually, an additional termination condition is specified.

Algorithm 2 (random search (policy))**.** Let $\mathcal{A}$ be a set of actions. Let $\mathcal{O}$ be a set of observations. Let $\Gamma \subseteq (\mathcal{O} \to \mathcal{A}) \to \mathbb{B}$ be a space of goal predicates on policy functions. Let $\gamma \in \Gamma$ be a goal predicate. We assume that the policy space $\Pi \subseteq \mathcal{O} \to \mathcal{A}$ can be sampled from, i.e., $\pi \sim \Pi$ returns a random element from Π. Random search for n samples is given via the function

$$\rho(n) = \begin{cases} \emptyset & \text{if } n = 0, \\ \pi & \text{if } n > 0 \text{ and } \gamma(\pi) \text{ where } \pi \sim \Pi, \\ \rho(n-1) & \text{otherwise.} \end{cases}$$

finding policies...