SMU Lyle School of Engineering

Reinforcement Learning

Reward Engineering

Sutton/Barto* Chapter 17.4
and other sources

Michael Hahsler, SMU

*Sutton and Barto, Reinforcement Learning: An Introduction, 2nd edition, MIT Press, Cambridge, MA, 2018

Online Material

CC BY NC SA

Topics of this Course

- troduction to reinforcement learning
- y decision processes
- bular Methods
  - rogramming
  - M methods
  - Temp nce learning
  - Multi-st pping
  - Planning a with tabular methods
- Part II: Approxi tion Methods
  - Prediction and Co Approximation
  - Eligibility Traces
  - Policy Gradient Metho
- Part III: Modern RL Metho
  - Deep Reinforcement Learning
  - Current Applications

Reward Engineering

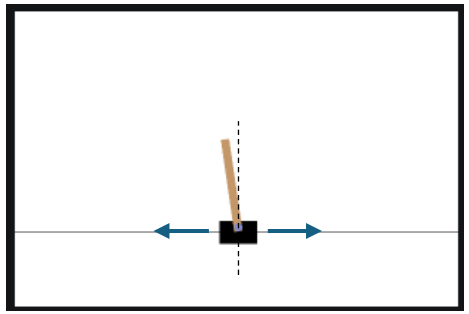
Problem Types and Reward

Types of RL Problems

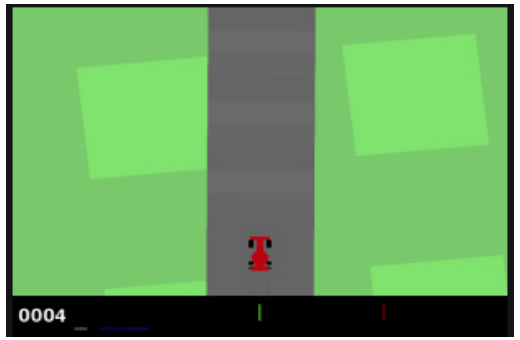
Robotics

AI
High-level decision making.
Control is done by smart actuators.

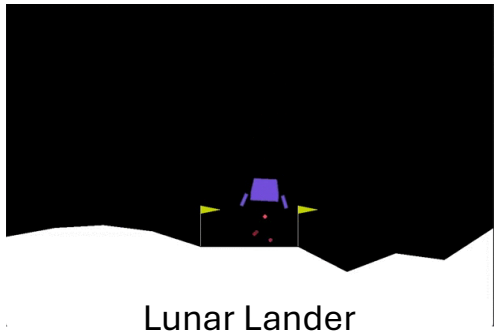
Control — **Control + Planning** — **Planning**



Inverted Pendulum/Cartpole



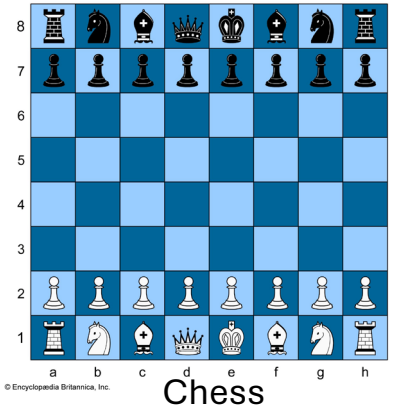
Car Racing



Lunar Lander

Frozen Lake Maze

Navigation Problem



Chess

Delayed (Sparse) Rewards

Designing Reward Signals

- **RL's advantage over supervised ML**:
  RL does not need the correct action to learn. It uses rewards instead.

- Remember the **Reward Hypothesis**: "All goals and purposes can be modeled as maximizing the expected cumulative sum of a received scalar signal (reward)."

- **Questions**:
  - How well does the reward signal frame the goal?
  - How well does the reward signal assess the progress of reaching the goal?

Often difficult to translate the goal to rewards.

- **Control problems**: Typically have direct and near immediate feedback.
- **Planning problems**:
  - **Sparse rewards** (e.g., only when a game is won). Leads to slow learning, excessive exploration, credit assignment problem.
  - Complex, **multi-step tasks**.
  - **"Supplemental" rewards** may hurt performance or lead to reward hacking.

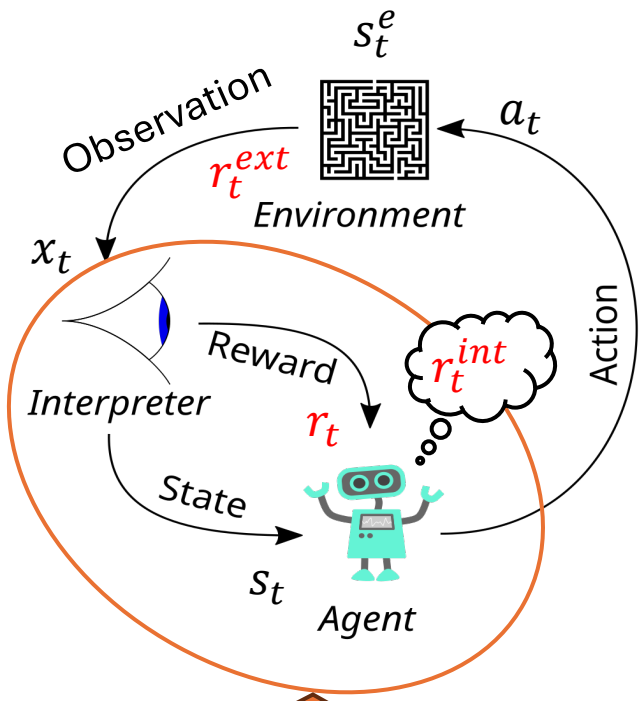
Reward Engineering

- Sometimes we separate:
  - External reward r^{ext} coming from the environment.
  - Internal reward r^{int} created by the agent.
- Both may be "engineered" by the creator of the task (environment) or the agent.

- The reward used for learning is then

$$r_t = r_t^{ext} + \beta r_t^{int}$$

- **Issues**:
  - **Shortcut problem**: Exploit superficial or spurious patterns (in the reward or transition model) rather than learning the actual task.
  - **Reward hacking**: Exploit loophole in reward structure.
  - **Overwriting**: Engineered reward overwriting the true objective.
  - **Bias problem**: Add (potentially wrong) assumptions of what optimal behavior should be.



The agent design often includes designing the **reward structure**

Reward Shaping Techniques

Potential-based Reward Shaping (PBRS)

- **Idea**:
  - States have a potential $\phi(s)$ representing how promising the state is to solve the task.
  - The agent can assess $\phi(s)$ and use the change in potential to modify the reward.

- The reward for a state transition $s \to s'$ is modified as:

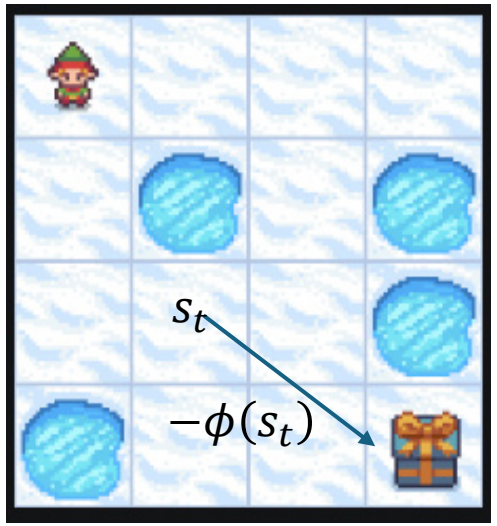
$$r'(s, a, s') = r(s, a, s') + \beta\big(\gamma\phi(s') - \phi(s)\big)$$

Scale so it does not overwrite the reward

- **Result**:
  - Densifies rewards with difference in potential.
  - Encourages the agent to learn to move to more promising state.
  - **Does not change the optimal policy.**

- Requirement: The agent needs a good heuristic for $\phi(s)$.

Example



Frozen Lake Maze

Use the negative distance to the goal as the potential function. Compare to A* search.

Andrew Y. Ng, Daishi Harada, and Stuart J. Russell. 1999. Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping. In Proceedings of the Sixteenth International Conference on Machine Learning (ICML '99).

Intermediate Milestone Rewards

- Add small one-time rewards for solving an important subtask:

$$r'(s, a, s') = r(s, a, s') + m_t$$

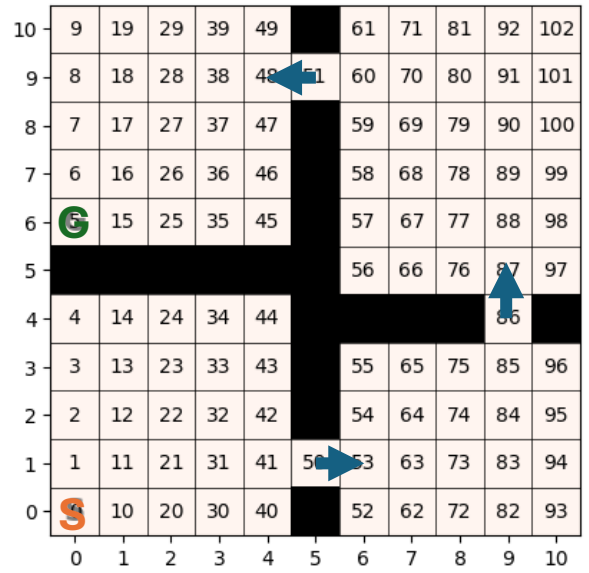
where $m_t = r_m$ when a milestone m is reached for the first time and 0 otherwise.

- Subtask completion can be represented as reaching a state for the first time. r_{m_1}

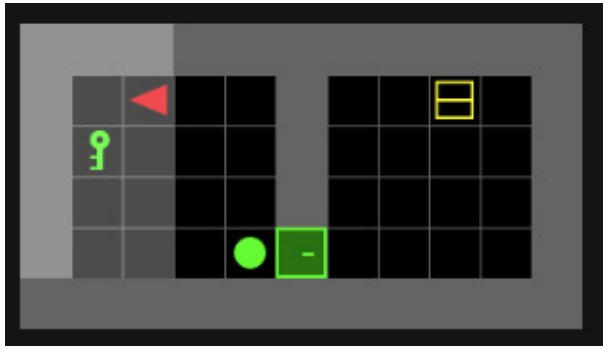
- This can help with complex, multi-step problems.
- **Note**: This can be seen as simplified version of PBRS.

- **Issues**:
  - Reward Hacking
  - Misleading Paths: If the intermediate reward is poorly designed

Examples milestones



Milestone is going to the next room



Milestones:
Picking up the key, opening the door.

Intrinsic Motivation: Curiosity

- Providing the agent with an internal reward bonus to encourage exploration.
- Modify reward by adding internal reward

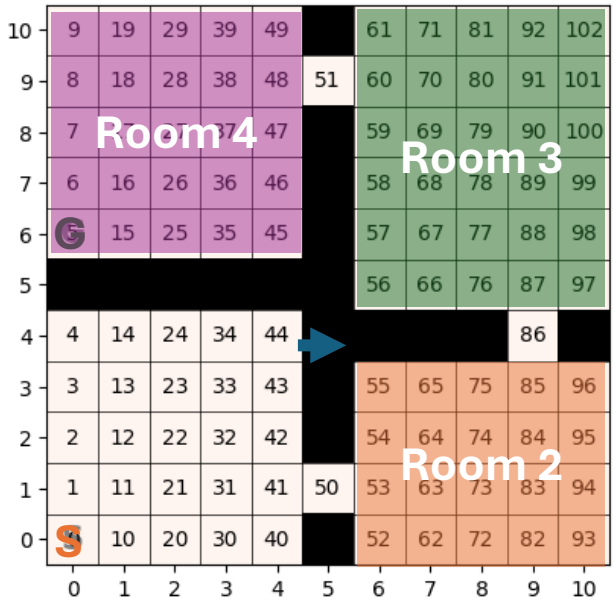
$$r_t = r_t^{ext} + \beta r_t^{int}$$

- r_t^{int} depends on level of novelty of the reached state.

- **How to measure novelty:**
  - Visit a new state/set of states for the first time.
  - Visit a state that has not been visited very often (the agent is uncertain about its value).
  - Visit a state that surprises the agent. I.e. the agent's current model produces a large prediction error (temporal difference error).

- **Issue**: "Noisy TV problem" agent gets trapped by the reward for staring at a static-filled, always changing unlearnable TV screen! Caused by state descriptions which contain highly stochastic elements.

Example



Use a reward bonus when new rooms are discovered.

Deepak Pathak, Pulkit Agrawal, Alexei A. Efros and Trevor Darrell. **Curiosity-driven Exploration by Self-supervised Prediction.** In *ICML 2017*.

Inverse RL

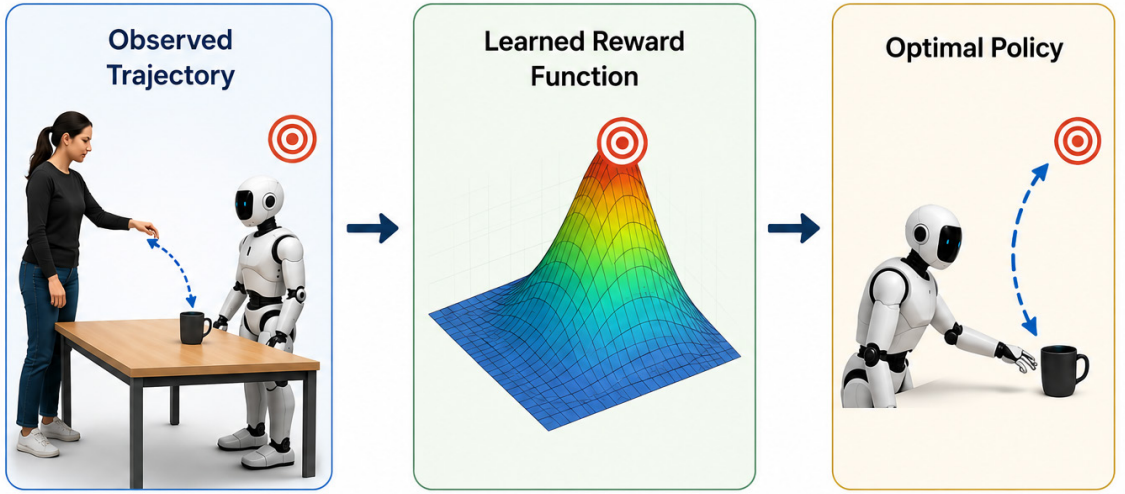
- **Idea**: Learn the hard to define reward function from demonstrations by an expert.
- **Input**: Expert behavior as sequences of (s, a, r, s') tuples .
- **Goal**: Finding a parameterized reward function $r^*(s, a, s')$ that explains the expert's behavior. I.e., that assigns the highest value to the expert's behavior.

$$\mathbb{E}\left[\sum_t \gamma^t r^*(s_t, a, s_{t+1}|\boldsymbol{w})|\pi^*\right] \geq \mathbb{E}\left[\sum_t \gamma^t r^*(s_t, a, s_{t+1}|\boldsymbol{w})|\pi\right] \quad \forall\pi$$

We assume the expert behaves optimally

- **Challenge**: The problem is "underspecified." For any given action, there are potentially infinite reward functions that could explain it. Therefore, IRL algorithms usually find the "most likely" or simplest reward function (using regularization) that fits the data.

- **Examples:**
  - **Autonomous Driving:** Observing how human drivers yield, navigate.
  - **Robotics:** Teaching robots complex physical tasks (like folding laundry or stacking objects) by showing them how to do it once, allowing the system to deduce the objective.
  - **Large Language Models (LLMs):** Used in AI alignment, where human feedback creates a reward model (via RLHF or similar structures) to align model behavior with human intent.



Andrew Y. Ng and Stuart J. Russell. 2000. Algorithms for Inverse Reinforcement Learning. In Proceedings of the Seventeenth International Conference on Machine Learning (ICML '00).

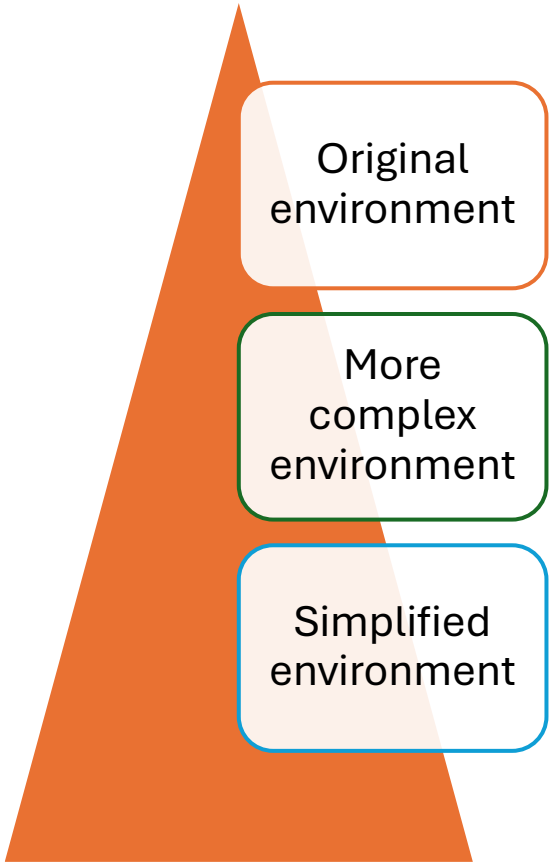
Related Training Techniques

Related Technique: Curriculum Learning

- **Goal**: Ensuring the agent masters foundational skills before moving on to more advanced challenges.

- **How It Works**:
  1. Train on an artificially simplified environment.
  2. As the agent's performance improve, increase the difficult stepwise by adding back more difficult challenges.
  3. End with an agent that has graduated from learning on the original, complex environment.

- **The Advantage**: The agent ultimately trains on the original problem. No reward hacking.



Weng, Lilian. (Jan 2020). Curriculum for reinforcement learning. Lil'Log.

Related Technique: Robust RL

- **Issue**: Standard RL often assumes the environment is fixed and perfectly known. This fails often:
  - Sim-to-Real Gap: Slight differences in friction, sensor noise, or wind result in performance drops when transitioning from a training simulator to a physical robot.
  - Overfitting: Standard agents often over-optimize for the exact distribution of training scenarios, performing poorly during unexpected changes.
- **Methods**:
  - **Robust Markov Decision Processes** (RMDPs): Instead of optimizing for a single, environment, the agent is trained on an uncertainty set. Identify a policy that performs well under the worst model in the set.
  - **Adversarial Training**: Modeled as a two-player zero-sum game. The agent aims to maximize rewards, while an environment adversary simultaneously attempts to tweak dynamics or observations to minimize the agent's reward.
  - **Conservative Smoothing & Regularization**: Mechanisms like Maximum Entropy RL introduce penalties for over-confident predictions, naturally forcing the agent to learn smoother, more adaptable policies.

S. Chen and Y. Li, "An Overview of Robust Reinforcement Learning," *2020 IEEE International Conference on Networking, Sensing and Control (ICNSC)*, Nanjing, China, 2020, pp. 1-6

What You Should Know

- Be able to explain how changing the reward structure affects RL:
  - How rewards express the goal.
  - Sparse rewards in planning problems.
- Understand the basic concepts of:
  - Extrinsic vs. extrinsic reward
  - Potential-based Reward Shaping (PBRS)
  - Intermediate Milestone Rewards
  - Intrinsic Motivation: Curiosity
  - Inverse RL
- Recognize issues
  - When the learned policy is the result of a shortcut or reward hacking.
  - When the engineered reward overwriting the true objective.
  - When the engineered reward has the potential to bias the solution.