Lecture 30: Markov Decision Processes

Correction: Constraints on Rational Preferences

- **Orderability**:$(A \succ B) \vee (B \succ A) \vee (A \sim B)$
- **Transitivity:** $(A \succ B) \wedge (B \succ C) \implies (A \succ C)$
- **Continuity:** $A \succ B \succ C \implies \exists p[p, A; 1-p, C] \sim B$
- **Substitutability:** $A \sim B \implies [p, A; 1-p, C] \sim [p, B; 1-p, C]$
- **Monotonicity:** $A \succ B \implies (p \geq q \iff [p, A; 1-p, B] \succsim [q, A; 1-q, B])$
- **Decomposability:**
  $[p, A; 1-p, [q, B; 1-q, C]] \sim [p, A; (1-p)q, B; (1-p)(1-q), C]$

Correction: Constraints on Rational Preferences

- **Orderability**:$(A \succ B) \vee (B \succ A) \vee (A \sim B)$
- **Transitivity:** $(A \succ B) \wedge (B \succ C) \implies (A \succ C)$
- **Continuity:** $A \succ B \succ C \implies \exists p[p, A; 1-p, C] \sim B$
- **Substitutability:** $A \sim B \implies [p, A; 1-p, C] \sim [p, B; 1-p, C]$
- **Monotonicity:** $A \succ B \implies (p \geq q \iff [p, A; 1-p, B] \succsim [q, A; 1-q, B])$
- **Decomposability:**
  $[p, A; 1-p, [q, B; 1-q, C]] \sim [p, A; (1-p)q, B; (1-p)(1-q), C]$

Correction: Constraints on Rational Preferences

Orderability: $(A \succ B) \vee (B \succ A) \vee (A \sim B)$

Isn't this too weak? This doesn't stop us from having $A \succ B$ **and** $B \succ A$

Correction: Constraints on Rational Preferences

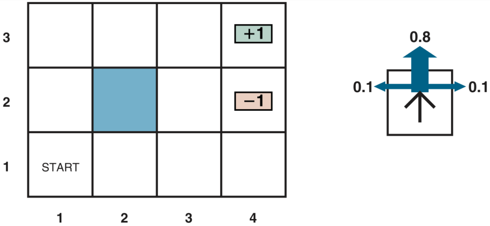
Orderability: Exactly one of $(A \succ B), (B \succ A)$, or $(A \sim B)$ must hold

Isn't this too weak? This doesn't stop us from having $A \succ B$ **and** $B \succ A$

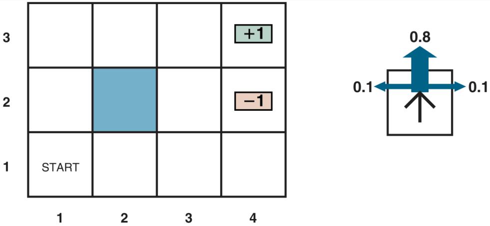
Last Lecture

Decision Making under Uncertainty for single-step problems

A Sequential Decision Problem



A Sequential Decision Problem

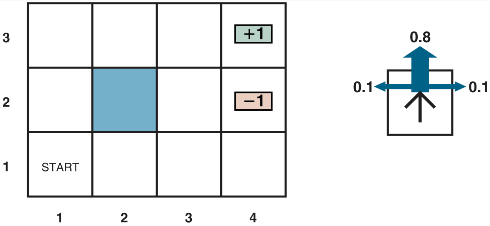


S_0 – initial state

$P(s'|s,a) = T(s,a,s')$ – transition model

$R(s)$ – reward function

A Sequential Decision Problem



S_0 – initial state

$P(s'|s, a) = T(s, a, s')$ – transition model

$R(s)$ – reward function

$$U_h(s_0, s_1, s_2, \dots)$$

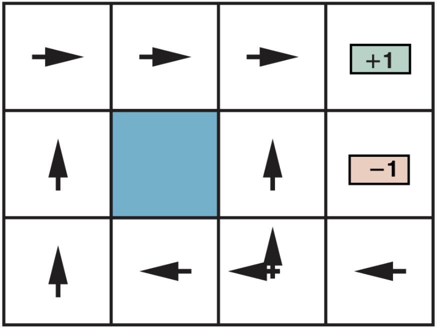
Markov Decision Process

$$MDP = \langle S_0, T(s, a, s'), R(s) \rangle$$

$\pi(s) -$ **policy**

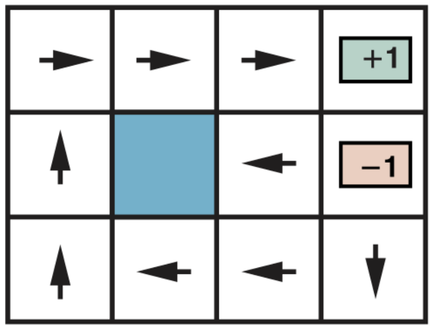
$\pi^*(s) -$ **optimal policy** (the one that yields highest expected utility)

Reward function affects optimal policy



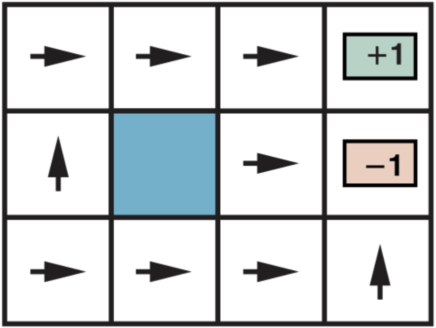
$$R(s) = -0.04$$

Reward function affects optimal policy



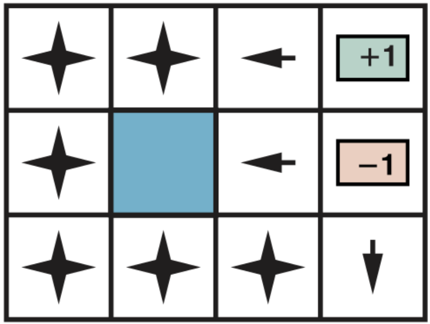
$$R(s) = -0.01$$

Reward function affects optimal policy



$R(s) = -1.15$

Reward function affects optimal policy



$R(s) = 2.0$

Finite vs Infinite Utilities

Infinite Horizon Utility Function:

$$U_h([s_1, s_2, \dots])$$

Finite Horizon Utility Function:

$$\forall k, U_h([s_1, s_2, \dots s_N, \dots, s_{N+k}]) = U_h([s_1, s_2, \dots s_N]) \quad \text{for fixed N}$$

Additive Reward and Infinite Horizons

$$U_h([s_0, s_1, s_2, \dots]) = R(s_1) + R(s_2) + \dots$$

Discounted Reward Formulation

$$
\begin{aligned}
U_h([s_0, s_1, s_2, \dots]) &= R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \gamma^3 R(s_3) + \dots \\
&= \sum_{t=0}^{\infty} \gamma^t R(s_t) \\
&\leq \sum_{t=0}^{\infty} \gamma^t R_{max} \\
&= \frac{R_{max}}{1-\gamma}
\end{aligned}
$$

Discounted Reward Formulation

$$
\begin{aligned}
U_h([s_0, s_1, s_2, \ldots]) &= R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \gamma^3 R(s_3) + \ldots \\
&= \sum_{t=0}^{\infty} \gamma^t R(s_t) \\
&\leq \sum_{t=0}^{\infty} \gamma^t R_{max} \\
&= \frac{R_{max}}{1-\gamma}
\end{aligned}
$$

Discounted Reward Formulation

$$
\begin{aligned}
U_h([s_0, s_1, s_2, \dots]) &= R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \gamma^3 R(s_3) + \dots \\
&= \sum_{t=0}^{\infty} \gamma^t R(s_t) \\
&\leq \sum_{t=0}^{\infty} \gamma^t R_{max} \\
&= \frac{R_{max}}{1-\gamma}
\end{aligned}
$$

Discounted Reward Formulation

$$
\begin{aligned}
U_h([s_0, s_1, s_2, \dots]) &= R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \gamma^3 R(s_3) + \dots \\
&= \sum_{t=0}^{\infty} \gamma^t R(s_t) \\
&\leq \sum_{t=0}^{\infty} \gamma^t R_{max} \\
&= \frac{R_{max}}{1 - \gamma}
\end{aligned}
$$

How good is a given state?

$$U^{\pi}(s) = E\left[\sum_{t=0}^{\infty} \gamma^t R(S_t)\right]$$

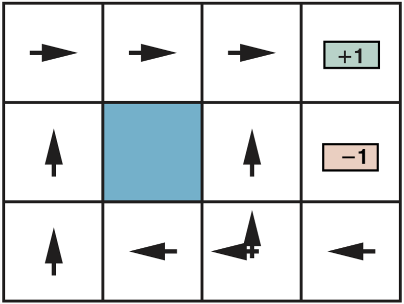
$$\pi_s^* = \underset{\pi}{argmax}\ U^{\pi}(s)$$

How good is a given state?

$$U^{\pi}(s) = E\left[\sum_{t=0}^{\infty} \gamma^{t} R(S_t)\right]$$

$$\pi_s^* = \underset{\pi}{argmax}\ U^{\pi}(s)$$

$\gamma = 1, R(s) = 0.04$



(a) $\pi(s)$

0.8516	0.9078	0.9578	+1
0.8016		0.7003	−1
0.7453	0.6953	0.6514	0.4279

(b) $U^{\pi}(s)$

Finding the Optimal Policy: The Bellman Equation

$$\pi^*(s) = \underset{\pi}{argmax}\ U^\pi(s)$$

$$\pi^*(s) = \underset{a}{argmax}\ \sum_{s'} P(s'|s,a)U^{\pi^*}(s')$$

Bellman Equation: $U^{\pi^*}(s) = R(s) + \gamma \max_a \sum_{s'} P(s'|s,a)U^{\pi^*}(s')$

Finding the Optimal Policy: The Bellman Equation

$$\pi^*(s) = \underset{\pi}{argmax}\ U^{\pi}(s)$$

$$\pi^*(s) = \underset{a}{argmax}\ \sum_{s'} P(s'|s,a) U^{\pi^*}(s')$$

Bellman Equation: $U^{\pi^*}(s) = R(s) + \gamma \max_a \sum_{s'} P(s'|s,a) U^{\pi^*}(s')$

Finding the Optimal Policy: The Bellman Equation

$$\pi^*(s) = \underset{\pi}{argmax}\ U^{\pi}(s)$$

$$\pi^*(s) = \underset{a}{argmax}\ \sum_{s'} P(s'|s,a)U^{\pi^*}(s')$$

Bellman Equation: $U^{\pi^*}(s) = R(s) + \gamma \max_a \sum_{s'} P(s'|s,a)U^{\pi^*}(s')$

Bellman Equations

Bellman Equation: $U^{\pi^*}(s) = R(s) + \gamma \max_a \sum_{s'} P(s'|s,a) U^{\pi^*}(s')$

Q: Problem with n states, how many bellman equations?

Bellman Equations

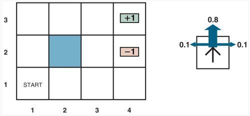
$$U^{\pi^*}(s_{\langle 1,1\rangle}) = R(s_{\langle 1,1\rangle}) + \gamma \max_a \sum_{s'} P(s'|s_{\langle 1,1\rangle}, a) U^{\pi^*}(s')$$

$$U^{\pi^*}(s_{\langle 1,2\rangle}) = R(s_{\langle 1,2\rangle}) + \gamma \max_a \sum_{s'} P(s'|s_{\langle 1,2\rangle}, a) U^{\pi^*}(s')$$

$$U^{\pi^*}(s_{\langle 1,3\rangle}) = R(s_{\langle 1,3\rangle}) + \gamma \max_a \sum_{s'} P(s'|s_{\langle 1,3\rangle}, a) U^{\pi^*}(s')$$

$$U^{\pi^*}(s_{\langle 1,4\rangle}) = R(s_{\langle 1,4\rangle}) + \gamma \max_a \sum_{s'} P(s'|s_{\langle 1,4\rangle}, a) U^{\pi^*}(s')$$

...



Bellman Equations

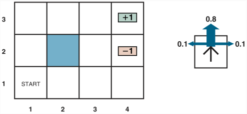
$$U^{\pi^*}(s_{\langle 1,1\rangle}) = \underbrace{R(s_{\langle 1,1\rangle})}_{\text{reward}} + \gamma \max_a \sum_{s'} \underbrace{P(s'|s_{\langle 1,1\rangle}, a)}_{\text{transition}} \underbrace{U^{\pi^*}(s')}_{\text{recursive}}$$

$$U^{\pi^*}(s_{\langle 1,2\rangle}) = R(s_{\langle 1,2\rangle}) + \gamma \max_a \sum_{s'} P(s'|s_{\langle 1,2\rangle}, a) U^{\pi^*}(s')$$

$$U^{\pi^*}(s_{\langle 1,3\rangle}) = R(s_{\langle 1,3\rangle}) + \gamma \max_a \sum_{s'} P(s'|s_{\langle 1,3\rangle}, a) U^{\pi^*}(s')$$

$$U^{\pi^*}(s_{\langle 1,4\rangle}) = R(s_{\langle 1,4\rangle}) + \gamma \max_a \sum_{s'} P(s'|s_{\langle 1,4\rangle}, a) U^{\pi^*}(s')$$

...



Value Iteration

Setup: Set Utility guesses to arbitrary values

$$U_0(s_{\langle 1,1 \rangle}) \leftarrow 0, U_0(s_{\langle 1,2 \rangle}) \leftarrow 0, \ldots$$

Repeat: One-step update Utilities using bellman equation

$$U_{i+1}(s_{\langle 1,1 \rangle}) \leftarrow R(s_{\langle 1,1 \rangle}) + \gamma \max_a \sum_{s'} P(s'|s_{\langle 1,1 \rangle}, a) U_i(s')$$

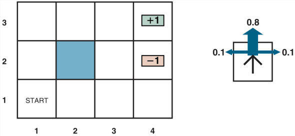
$$\ldots$$

$$U_{i+1}(s_{\langle 3,4 \rangle}) \leftarrow R(s_{\langle 3,4 \rangle}) + \gamma \max_a \sum_{s'} P(s'|s_{\langle 3,4 \rangle}, a) U_i(s')$$

Until: $U_{i+1}(s) \approx U_i(s)$ **for all** s

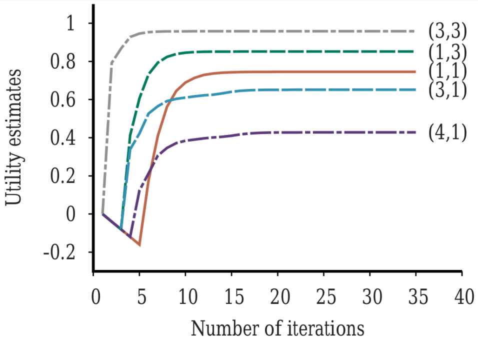
Example Update

$$
\begin{aligned}
&U_{i+1}(s_{\langle 1,1\rangle}) \leftarrow -0.04 + \gamma \max[\\
&\quad 0.8U_i(s_{\langle 1,2\rangle}) + 0.1U_i(s_{\langle 2,1\rangle}) + 0.1U_i(s_{\langle 1,1\rangle}), && \textit{(up)} \\
&\quad 0.9U_i(s_{\langle 1,1\rangle}) + 0.1U_i(s_{\langle 1,2\rangle}), && \textit{(left)} \\
&\quad 0.9U_i(s_{\langle 1,1\rangle}) + 0.1U_i(s_{\langle 2,1\rangle}), && \textit{(down)} \\
&\quad 0.8U_i(s_{\langle 2,1\rangle}) + 0.1U_i(s_{\langle 1,2\rangle}) + 0.1U_i(s_{\langle 1,1\rangle}) && \textit{(right)} \\
&]
\end{aligned}
$$



Value Iteration in Action

0.8516	0.9078	0.9578	+1
0.8016		0.7003	−1
0.7453	0.6953	0.6514	0.4279



Summary

- Sequential decision making
- Markov Decision Processes
- Value Iteration

Returning a the Map of the Landscape

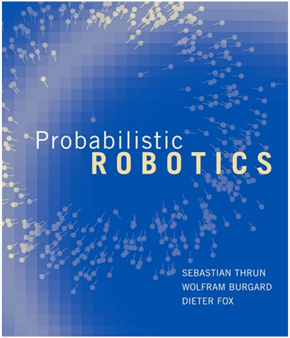
Scraped from Lecture 1: (Kwabena Nuamah)

- **"Benign"**: Fully Observable, Deterministic, Episodic, Static, Discrete and Single Agent
- **"Chaotic"**: Partially Observable, Stochastic, Sequential, Dynamic, Continuous, Multi-agent

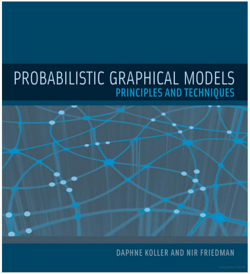
Continuous, Stochastic, Partially Observable, Sequential, . . .

Courses

- Introduction to Mobile Robotics (MOB)
- Advanced Robotics



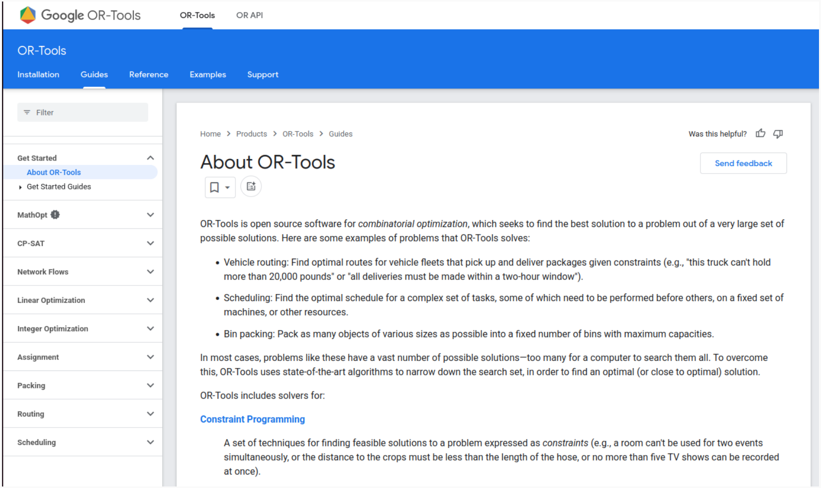
Graphical Models



Courses:

- Probabilistic Modelling and Reasoning (PMR)
- Methods for Causal Inference (MCI)

I liked Constraint Solving / Coursework 1!



Google OR-Tools

OR-Tools OR API

OR-Tools

Installation Guides Reference Examples Support

Filter

Get Started
About OR-Tools
Get Started Guides
MathOpt
CP-SAT
Network Flows
Linear Optimization
Integer Optimization
Assignment
Packing
Routing
Scheduling

Home > Products > OR-Tools > Guides

Was this helpful?

About OR-Tools

Send feedback

OR-Tools is open source software for *combinatorial optimization*, which seeks to find the best solution to a problem out of a very large set of possible solutions. Here are some examples of problems that OR-Tools solves:

- Vehicle routing: Find optimal routes for vehicle fleets that pick up and deliver packages given constraints (e.g., "this truck can't hold more than 20,000 pounds" or "all deliveries must be made within a two-hour window").
- Scheduling: Find the optimal schedule for a complex set of tasks, some of which need to be performed before others, on a fixed set of machines, or other resources.
- Bin packing: Pack as many objects of various sizes as possible into a fixed number of bins with maximum capacities.

In most cases, problems like these have a vast number of possible solutions—too many for a computer to search them all. To overcome this, OR-Tools uses state-of-the-art algorithms to narrow down the search set, in order to find an optimal (or close to optimal) solution.

OR-Tools includes solvers for:

Constraint Programming

A set of techniques for finding feasible solutions to a problem expressed as *constraints* (e.g., a room can't be used for two events simultaneously, or the distance to the crops must be less than the length of the hose, or no more than five TV shows can be recorded at once).

I liked PDDL / Coursework 2!

International Conference on Automated Planning and Scheduling

Menu ••• | Search

Competitions

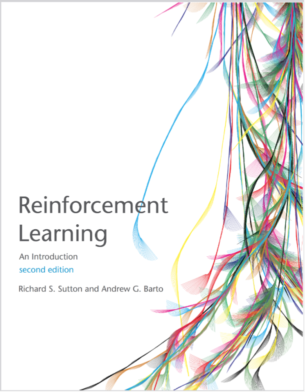
The International Planning Competition 2023

- Classical Tracks
  - Daniel Fišer, Saarland University
  - Florian Pommerening, University of Basel
- Learning Tracks
  - Jendrik Seipp, Linköping University
  - Javier Segovia-Aguas, Universitat Pompeu Fabra
- Probabilistic Tracks
  - Ayal Taitler, University of Toronto
  - Scott Sanner, University of Toronto
- Numeric Tracks
  - Joan Espasa Arxer, University of St Andrews
  - Enrico Scala, University of Brescia
- HTN Tracks
  - Ron Alford, MITRE
  - Dominik Schreiber, Karlsruhe Institute of Technology
  - Gregor Behnke, University of Amsterdam

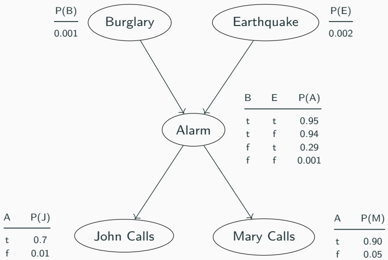
Wait! What was all that stuff about MDPs?

Courses

- Reinforcement Learning (RL)



Where did these parameters come from?



Courses

- Machine Learning (MLG)
- Introductory Applied Machine Learning (IAML)
- Machine Learning Practical (MLP)
- Machine Learning Systems (MLS)
- Machine Learning Theory (MLT)
- Machine Learning and Pattern Recognition (MLPR)

I can make an AI agent...but should I?