Planning with a model you don't Trust

When can we still use dynamic programming under model uncertainty?

Axel Benyamine

CMAP, École polytechnique · Inria Paris

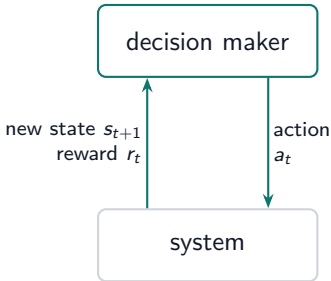
joint work with Alain Durmus, Julien Grand-Clément, Marek Petrik, and Michael I. Jordan

Dynamic Programming for Epistemic Uncertainty in Markov Decision Processes, ICML 2026 (Spotlight)

Sequential decisions under uncertainty

Many (industrial) problems share one structure:

- treating a patient over months,
- managing an inventory or an energy grid,
- routing a fleet of vehicles,
- deciding what a robot, or an ad system, does next.



Every period: observe the **state**, pick an **action**, collect a **reward**, the system moves to a random new state. Repeat.

Goal: a **rule** that does well over the long run, not only today.

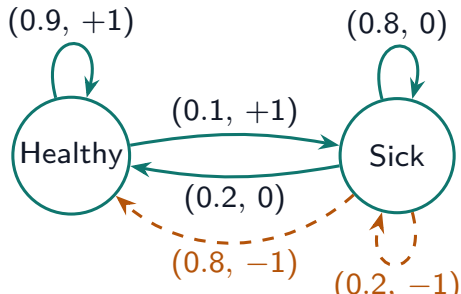
A Markov Decision Process (MDP)

- **states** $s \in \mathcal{S}$: Healthy, Sick
- **actions** $a \in \mathcal{A}$: wait, treat
- **transitions** $P(s, a, s')$: where do we go next?
- **rewards** $r(s, a, s')$: $+1$ if healthy, -1 to treat
- **discount** $\gamma < 1$: tomorrow counts γ times less

Policy π: a rule *state* $\mapsto$ *action*.

Goal: maximise the total discounted reward

$$\sum_{t \geq 0} \gamma^t \, r(\widetilde{S}_t, \widetilde{A}_t, \widetilde{S}_{t+1}).$$



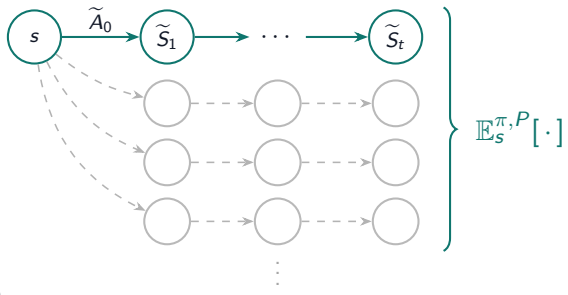
Randomness inside the model (Aleatoric uncertainty)

Even with P and π fixed, **the trajectory is random**: this month the patient may or may not fall sick.

Average it out: the **value function**

$$V^{\pi,P}(s) = \mathbb{E}_s^{\pi,P}\Big[\sum_{t\geq 0}\gamma^t r(\widetilde{S}_t, \widetilde{A}_t, \widetilde{S}_{t+1})\Big]$$

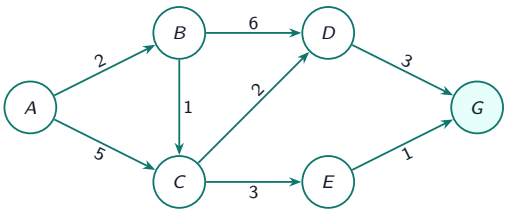
is *one number per state*: "how good is it to be in s if I follow π?"



one model, many trajectories

Optimal value: the best achievable from s, $V^P(s) = \sup_\pi V^{\pi,P}(s)$.
"how good is it to be in s if I play optimally afterwards?"

Dynamic programming: the fastest route from A to G



roads: travel time **boxes**: time-to-go to G

Bellman's principle of optimality

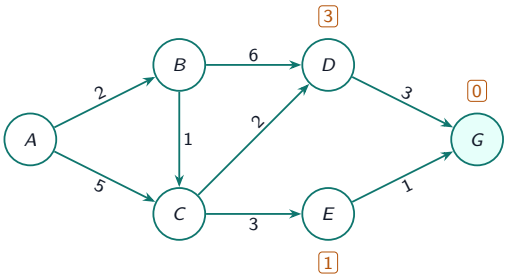
If the best route from A to G goes through C, its tail from C to G is the best route from C.

So the *time-to-go* d satisfies

$$d(x) = \min_{y \text{ next to } x} \big[\,\text{time}(x \to y) + d(y)\,\big].$$

- **solve backwards** from the goal, one node at a time;
- the answer is a **table**: at each node, which way to go, whatever the past.

Dynamic programming: the fastest route from A to G



roads: travel time **boxes**: time-to-go to G

Bellman's principle of optimality

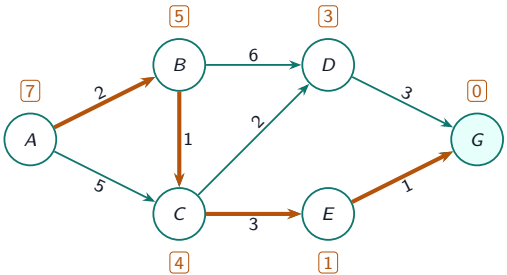
If the best route from A to G goes through C, its tail from C to G is the best route from C.

So the *time-to-go* d satisfies

$$d(x) = \min_{y \text{ next to } x} \big[\text{time}(x \to y) + d(y)\big].$$

- **solve backwards** from the goal, one node at a time;
- the answer is a **table**: at each node, which way to go, whatever the past.

Dynamic programming: the fastest route from A to G



roads: travel time **boxes**: time-to-go to G

Bellman's principle of optimality

If the best route from A to G goes through C, its tail from C to G is the best route from C.

So the *time-to-go* d satisfies

$$d(x) = \min_{y \text{ next to } x} \big[\,\text{time}(x \to y) + d(y)\,\big].$$

- **solve backwards** from the goal, one node at a time;
- the answer is a **table**: at each node, which way to go, whatever the past.

Dynamic programming for MDPs

One change: the next node is chosen $\rightarrow$ the next state is random: *average* over s'

Bellman equation: value today = best action's [reward now + $\gamma\times$ average value tomorrow]

$$V(s) = \max_{a \in \mathcal{A}} \sum_{s'} P(s, a, s')(r(s, a, s') + \gamma\, V(s'))$$

Theorem **(Bellman, 1957; Puterman, 1994)**

The optimal value V^P is the unique solution. Consequences:

- **fast algorithms**: apply the equation repeatedly, the error shrinks by γ at each step;
- an optimal policy is a **lookup table** $s \mapsto a$, optimal from every starting state.

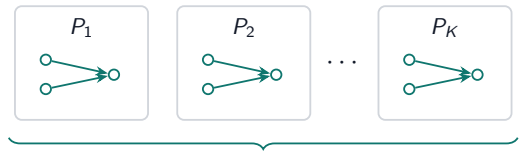
But who gives us P?

In practice P is **estimated** from data:
clinical trials, logs, a simulator.

$\widehat{P} \neq P$: a plan optimised for $\widehat{P}$
may do badly under the true P.

Two very different kinds of uncertainty:

- **Aleatoric**: the dice inside a known model.
  A fair coin lands heads or tails.
- **Epistemic**: *which* model?
  We do not know whether the coin is fair.



Epistemic uncertainty shrinks with more data.
Aleatoric uncertainty never does.

Robust decisions: we make them every day

- **Wearing a seatbelt**: a small cost on every trip, a payoff only on the worst one.
- **Safety factors**: a bridge is designed for loads far beyond the expected traffic.
- A cash reserve, a spare tyre, an umbrella in the bag.

The common pattern:

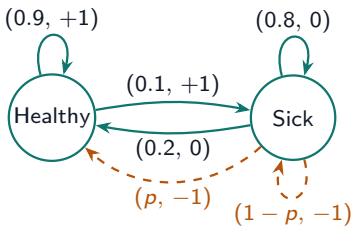
Choose the decision whose **worst plausible outcome** is the most acceptable.

Robust MDP: maximise the worst-case value over the plausible models

$$\sup_{\pi} \inf_{P \in \mathcal{P}} V^{\pi,P}$$

$\mathcal{P}$: a confidence region built from the data.

Robust MDPs on the health example



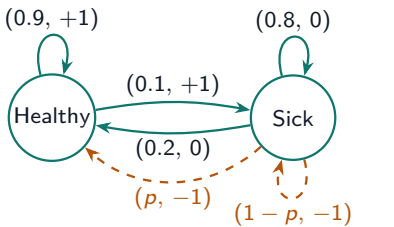
Value of being Sick ($\gamma = 0.9$):

p	0.5	0.6	0.7	**0.8**	0.9
wait	4.86	4.86	4.86	4.86	4.86
treat	4.06	4.79	5.37	5.82	6.20

The treatment succeeds with unknown probability p.

Best estimate $\widehat{p} = 0.8$; with few patients, all we can trust is $p \in [0.5, 0.9]$.

Robust MDPs on the health example



wait treat (probability, reward)

The treatment succeeds with unknown probability p.

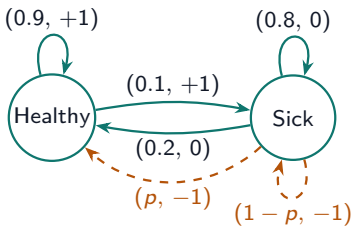
Best estimate $\widehat{p} = 0.8$; with few patients, all we can trust is $p \in [0.5, 0.9]$.

Value of being Sick ($\gamma = 0.9$):

p	0.5	0.6	0.7	**0.8**	0.9
wait	4.86	4.86	4.86	4.86	4.86
treat	4.06	4.79	5.37	**5.82**	6.20

- **Nominal plan**, trusting $\widehat{p} = 0.8$: treat ($5.82 > 4.86$).

Robust MDPs on the health example



wait treat (probability, reward)

The treatment succeeds with unknown probability p.

Best estimate $\widehat{p} = 0.8$; with few patients, all we can trust is $p \in [0.5, 0.9]$.

Value of being Sick ($\gamma = 0.9$):

p	0.5	0.6	0.7	**0.8**	0.9
wait	4.86	4.86	4.86	4.86	4.86
treat	**4.06**	4.79	5.37	**5.82**	6.20

- **Nominal plan**, trusting $\widehat{p} = 0.8$: treat ($5.82 > 4.86$).
- **Robust plan**, ready for any $p \in [0.5, 0.9]$: treat guarantees only 4.06, wait guarantees 4.86 $\Rightarrow$ wait.

The price of safety: we give up $5.82 \to 4.86$ if $\widehat{p}$ was right.

Robust dynamic programming

Put the worst case *inside* the Bellman equation, state by state:

$$V(s) = \max_{a \in \mathcal{A}} \min_{P_s \in \mathcal{P}_s} \sum_{s'} P(s,a,s')(r(s,a,s') + \gamma\, V(s'))$$

Theorem **(Wiesemann et al. (2013))**

Robust Bellman equations hold (with some assumptions on $\mathcal{P}$).

Good: same algorithms, same lookup-table policies as a classical MDP.

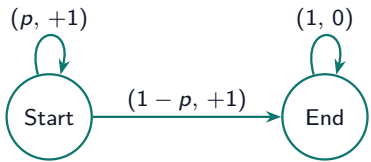
Bad: very conservative.

Can we be less pessimistic *and* keep dynamic programming?

Can we be less pessimistic?

Natural, softer criteria:

- **average** over the plausible models
- **percentile**: a guarantee valid for 95% of the models
- **CVaR**: the average over the 5% worst models



one action; p uniform on $[0, 1]$, drawn once; $\gamma = \frac{1}{2}$

$$V(\text{Start}) = \mathbb{E}_p\left[\frac{1}{1-\gamma p}\right] = 2\log 2 \approx 1.39$$

$$\text{Bellman:} \quad 1 + \gamma\,\mathbb{E}[p]\,V(\text{Start}) \approx 1.35 \quad \neq$$

One action, nothing to decide, and the Bellman equation already fails.

One language to rule them all: ambiguity-averse MDPs

1. Model the unknown transitions as a **random variable** $\widetilde{P} \sim \nu$ (e.g. a posterior given the data).
2. The return of a policy becomes a random variable: $V^{\pi,\widetilde{P}}$.
3. Summarise it with a **risk measure** ρ and solve $\sup_{\pi} \rho\big(V^{\pi,\widetilde{P}}\big)$.

Same Bellman operators, with ρ in place of the average over models.
When the Bellman equations hold: fast algorithms and lookup-table policies.

Characterising the class of problems where Dynamic Programming works

So: which ρ?

ρ	reads as	model	DP?
ess inf	worst plausible model	robust MDP	**yes**
ess sup	best plausible model	optimistic MDP	**yes**
$\mathbb{E}$	average model	multi-model MDP	**no**
VaR_α	a quantile of the return	percentile optimisation	**no**
Anything else?			**?**

Characterising the class of problems where Dynamic Programming works

So: which ρ?

ρ	reads as	model	DP?
ess inf	worst plausible model	robust MDP	**yes**
ess sup	best plausible model	optimistic MDP	**yes**
$\mathbb{E}$	average model	multi-model MDP	**no**
VaR_α	a quantile of the return	percentile optimisation	**no**
Anything else?			**no**

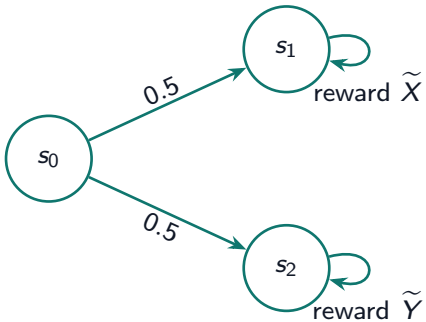
our result

Takeaways

- **MDP**: states, actions, random transitions, rewards. A policy is a rule *state* $\mapsto$ *action*.
- **Dynamic programming**: solve backwards from the goal.
  Good guarantees (fast algorithms, . . .)
- **Epistemic uncertainty**: the model itself is uncertain.
- **Robust MDPs** are the seatbelt: optimize for the worst plausible model.
  Keeps the whole dynamic-programming machinery.
- **Anything softer** (average, percentile, CVaR, . . .) leaves dynamic programming:
  Need for more complex algorithms and policies that must remember the past. Ways out: augment the state, policy gradient, nested risk measures.

Dynamic Programming for Epistemic Uncertainty in Markov Decision Processes, ICML 2026

Backup: one tiny instance already forces additivity



one action; $\widetilde{X}, \widetilde{Y}$ independent uncertain rewards

$$V(s_0) \approx \rho\Big(\frac{\widetilde{X}+\widetilde{Y}}{2}\Big)$$

$$TV(s_0) \approx \rho\Big(\frac{\rho(\widetilde{X})+\rho(\widetilde{Y})}{2}\Big) = \frac{\rho(\widetilde{X})+\rho(\widetilde{Y})}{2}$$

$V = TV$ **forces**

$$\rho(\widetilde{X}+\widetilde{Y}) = \rho(\widetilde{X}) + \rho(\widetilde{Y})$$

Richer branches give positive homogeneity (rules out entropic risk) and multiplicativity, which together leave only ess inf, ess sup (and $\mathbb{E}$ for resampled kernels).