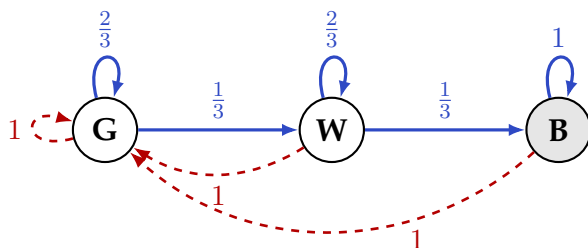


Lecture 4: Optimal Values and Policies

How to use these notes. Board content in board order: figures, equations, tables, one-line statements. Today's model is the machine of Lecture 3. The first section restates what this note uses from that one: the values of a fixed policy with their Bellman equation and its linear solve, and the values of three policies. The quiz takes the first fifteen minutes, so the meeting is sixty minutes long and this note is fuller than the meeting. Reading: [Sutton and Barto \(2018, Section 3.6\)](#): optimal value functions, the Bellman optimality equation, the backup diagrams of Figure 3.4, Examples 3.8 and 3.9; Section 4.4 there prepares value iteration. Optional: [Puterman \(1994, Section 6.2\)](#) for the existence theorem with its proof, and [Agarwal et al. \(2022, Section 1.1.3\)](#) for the same material in Q form (Theorems 1.7 and 1.8). Throughout the course, the notes draw on the public materials of Stanford CS 234 ([Brunskill, 2026](#)), UIUC CS 443 ([Jiang, 2026](#)), and Cornell CS 4789 ([Sun, 2025](#)); some examples are adapted from them.

Lecture 3 Left a Comparison

The machine again: good, worn, or broken; run or repair; $\gamma = 0.9$; it starts good.



$r(s, a)$	run	repair
G	8	-10
W	4	-10
B	0	-10

Where each action leads. Solid: run. Dashed: repair, always back to G.

This week's profit, fixed once (s, a) is.

From Lecture 3: the values of a fixed policy π . $V^\pi(s)$ is the expected discounted return of π started at s ; $Q^\pi(s, a)$ is the same with the first action forced to a . For every s and every (s, a) ,

$$V^\pi(s) = \sum_a \pi(a | s) \left[r(s, a) + \gamma \sum_{s'} P(s' | s, a) V^\pi(s') \right], \quad (\text{E}_\pi)$$

$$Q^\pi(s, a) = r(s, a) + \gamma \sum_{s'} P(s' | s, a) V^\pi(s'), \quad V^\pi(s) = \sum_a \pi(a | s) Q^\pi(s, a), \quad (\text{Q}_\pi)$$

$$Q^\pi(s, a) = r(s, a) + \gamma \sum_{s'} P(s' | s, a) \sum_{a'} \pi(a' | s') Q^\pi(s', a'). \quad (\text{Q}'_\pi)$$

In vectors, (E_π) is $V^\pi = r^\pi + \gamma P^\pi V^\pi$ with $r^\pi(s) = \sum_a \pi(a | s) r(s, a)$ and $P^\pi(s' | s) = \sum_a \pi(a | s) P(s' | s, a)$. For $\gamma < 1$, $I - \gamma P^\pi$ is invertible, so (E_π) has exactly one solution, $V^\pi = (I - \gamma P^\pi)^{-1} r^\pi$: any vector that satisfies (E_π) is V^π . One iteration of its right-hand side is the operator $(\mathcal{T}^\pi V)(s) := \sum_a \pi(a | s) [r(s, a) + \gamma \sum_{s'} P(s' | s, a) V(s')]$, so (E_π) reads $V^\pi = \mathcal{T}^\pi V^\pi$.

- Lecture 3 evaluated three of the eight deterministic policies, one linear solve of (E_π) each:

$\gamma = 0.9$	$V^\pi(\text{G}) = J(\pi)$	$V^\pi(\text{W})$	$V^\pi(\text{B})$
never repair	27.5	10	0
repair when broken	44.30	32.41	29.87
repair when worn (and when broken)	38.46	24.62	24.62

- Under “never repair”, $Q^\pi(s, \text{repair}) = -10 + 0.9 \cdot 27.5 = 14.75$ in every row. At W and at B one repair beats the policy’s own action ($14.75 > 10$, $14.75 > 0$); at G it does not ($14.75 < 27.5$). Lecture 3 stopped there.
- Lecture 1, on the map: $V^*(S) = \min\{7, 6.2\}$, the better of two roads, each priced by what follows. That was an optimality equation, with costs and a min, before we had the words for it.

What does “the best policy” mean when every state has its own number, and what equation do the best numbers satisfy?

Today’s targets.

- Define V^* and Q^* as per-state suprema over all rules; state the existence theorem (one deterministic stationary policy attains every supremum at once) and what it does not say.
- Derive the Bellman optimality equation for V^* from the definition alone: every rule splits at its first step, no rule beats one lookahead against V^* , and a rule can be built that matches it. Write the Q^* version, then prove the theorem from the two in three lines: the greedy table is optimal. Write the machine’s three equations and solve those by hand: guess, solve, verify. Name the operator $\mathcal{T}$.
- Extract π^* from V^* by a one-step lookahead, from Q^* by a table lookup: the table returned is the one the proof built, and every optimal table arises this way.
- Test a policy for optimality by checking its own values against the equation: “never repair” fails, “repair when broken” passes, and no other policy has to be evaluated.
- Recognize the failure mode: greedy with respect to values that are not V^* .

The Best Policy

- A deterministic policy is a table, one action per state: $|\mathcal{A}|^{|\mathcal{S}|}$ of them, $2^3 = 8$ for the machine, $4^9 = 262,144$ for Lecture 2's Gridworld. Stochastic policies: infinitely many.
- Brute force: one linear solve per table. Eight is fine; 262,144 is not; and nothing in the eight answers says which table to try next in a bigger problem.

$\pi(G), \pi(W), \pi(B)$		$V^\pi(G)$	$V^\pi(W)$	$V^\pi(B)$
run, run, run	never repair	27.5	10	0
run, run, repair	repair when broken	44.30	32.41	29.87
run, repair, run		38.46	24.62	0
run, repair, repair	repair when worn	38.46	24.62	24.62
repair, run, run		-100	10	0
repair, run, repair		-100	-65	-100
repair, repair, run		-100	-100	0
repair, repair, repair	always repair	-100	-100	-100

All eight deterministic policies at $\gamma = 0.9$, one linear solve each. One row is largest in every column.

- Repairing a good machine restarts the repair bill forever: -100 from G, whatever the policy does elsewhere; four of the eight rows are ruled out by their first entry.
- Comparing policies: π is at least as good as π' if $V^\pi(s) \geq V^{\pi'}(s)$ for every s (Sutton and Barto, 2018, Section 3.6). A partial order: (repair, run, repair) and (repair, repair, run) are incomparable, W prefers the first and B the second. Yet the table has a top row: (run, run, repair) is at least as good as every other row in every state.

Optimal values, optimal policy

$$V^*(s) := \sup_{\pi} V^\pi(s), \quad Q^*(s, a) := \sup_{\pi} Q^\pi(s, a),$$

for each s and each (s, a) separately, over every *rule* for choosing actions: a table or not, stochastic or not, using the clock or the past or not; $V^\pi(s)$ and $Q^\pi(s, a)$ are, as for a table, the expected discounted returns of the rule started at s , with the first action free or forced to a . sup: the least upper bound, a max once some rule attains it (the theorem below). A policy π^* is *optimal* if $V^{\pi^*}(s) = V^*(s)$ for every s : no rule beats it anywhere.

- Per state. Nothing in the definition says that one rule wins every state, or that any rule attains the sup. The theorem below settles both; it is proved right after the equation.
- Optimal in every state is optimal for every start: for every d_0 ,

$$J(\pi^*) = \sum_s d_0(s) V^*(s) \geq \sum_s d_0(s) V^\pi(s) = J(\pi),$$

so π^* does not depend on where the process begins. The converse fails: a policy that is best for one start may act badly in states that start never reaches.

Theorem (an optimal table exists). Finite $\mathcal{S}$ and $\mathcal{A}$, $\gamma < 1$. There is a deterministic stationary policy π^* with

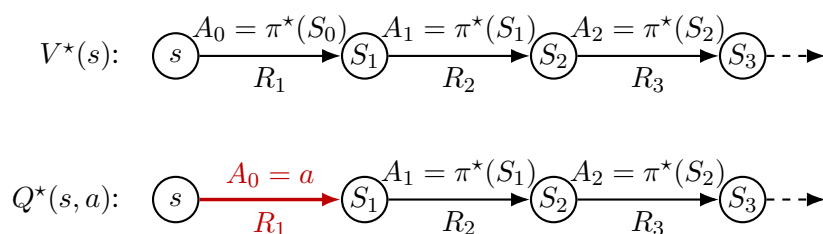
$$V^{\pi^*}(s) = V^*(s) \text{ for every } s, \quad Q^{\pi^*}(s, a) = Q^*(s, a) \text{ for every } (s, a).$$

Statement as in [Puterman \(1994, Section 6.2\)](#) and [Agarwal et al. \(2022, Theorem 1.7\)](#). Proof on page 6, right after the optimality equation and its Q^* form: three lines from those and the one solution of (E_π) . The derivation of the equation does not use the theorem.

- One table attains every sup at once, so “the optimal policy” makes sense, $\sup = \max$, and $V^* = V^{\pi^*}$, $Q^* = Q^{\pi^*}$ are the values of an ordinary policy: (E_π) , (Q_π) , and the linear solve apply to them.
- Deterministic and stationary: no coin, no clock, no memory needed, however the problem is posed. Not unique: two tables can share V^* (a tie between two actions somewhere). V^* and Q^* are unique: they are suprema.
- Why one table can suffice: the Markov property. What happens from s' on depends on s' alone, not on how the process got there, so the best behavior from s' is the same whether we started at G or at B. The derivation below writes this sentence as an equation, $(*)$, and its lower bound puts it to work: the rule from s' on may be chosen at s' .

Example. The machine at $\gamma = 0$: what is optimal?

Only this week counts, so $V^*(s) = \max_a r(s, a)$: run everywhere, $V^* = (8, 4, 0)$, “never repair”. At $\gamma = 0.9$ the top row repairs a broken machine. Same machine, two optimal policies: γ is part of the problem (Lecture 2’s Stop/Go).



Who chooses what. V^* : an optimal policy picks every action. Q^* : the first action is ours, forced; optimal behavior resumes after it.

- $Q^*(s, a)$ answers “what is a worth here if I act as well as possible afterwards?” It can be below $V^*(s)$: a forced first step can be a mistake that the best continuation only partly repairs.

The Bellman Optimality Equation

(E_π) averages over the policy's action. The best rule's first step is a choice, and a choice is a max.

Derivation. Fix a state s . Every rule π started at s splits at its first step: an action now, then a rule from wherever the machine lands. The first-step split behind (E_π) , with the continuation rule's value in place of $V^\pi(s')$:

$$V^\pi(s) = \sum_a \pi(a | s) \left[r(s, a) + \gamma \sum_{s'} P(s' | s, a) V^{\pi_{s'}}(s') \right]. \quad (*)$$

Here $\pi_{s'}$ is the rule π turns into after doing a and landing in s' : again a rule, started at s' . Markov: from s' on, the future depends on the past only through what the rule does next. For a table, $\pi_{s'} = \pi$ and $(*)$ is (E_π) .

Upper bound: no rule beats one lookahead against V^* .

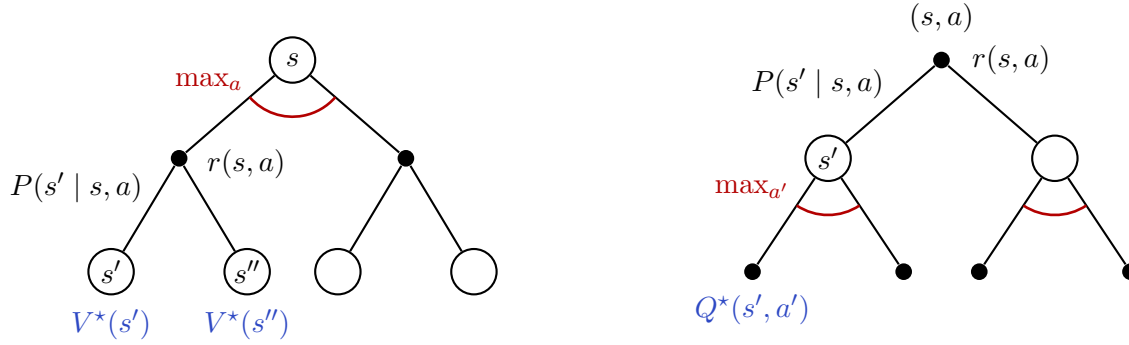
$$\begin{aligned} V^\pi(s) &\leq \sum_a \pi(a | s) \left[r(s, a) + \gamma \sum_{s'} P(s' | s, a) V^*(s') \right] && (*); V^{\pi_{s'}}(s') \leq V^*(s') \\ &\leq \max_a \left[r(s, a) + \gamma \sum_{s'} P(s' | s, a) V^*(s') \right] && \text{average} \leq \text{largest term} \\ V^*(s) &\leq \max_a \left[r(s, a) + \gamma \sum_{s'} P(s' | s, a) V^*(s') \right] && \text{every rule, so their sup} \end{aligned}$$

Lower bound: a rule can match one lookahead against V^* . Pick any action a and, for each s' , any rule $\pi_{s'}$. "Do a now; after landing in s' , follow $\pi_{s'}$ " is a rule, and $(*)$ prices it:

$$\begin{aligned} V^*(s) &\geq r(s, a) + \gamma \sum_{s'} P(s' | s, a) V^{\pi_{s'}}(s') && \text{no rule beats } V^* \\ V^*(s) &\geq r(s, a) + \gamma \sum_{s'} P(s' | s, a) V^*(s') && \text{sup over each } \pi_{s'} \\ V^*(s) &\geq \max_a \left[r(s, a) + \gamma \sum_{s'} P(s' | s, a) V^*(s') \right] && \text{every } a \end{aligned}$$

The bounds meet (Bellman, 1957). For every $s \in \mathcal{S}$:

$$\boxed{V^*(s) = \max_a \left[r(s, a) + \gamma \sum_{s'} P(s' | s, a) V^*(s') \right]} \quad (1)$$



V^* : a max over a at the root, then an average over s' . Q^* : an average over s' , then a max over a' at each s' .

The same for Q^* . Fix the first action at a , so $(*)$ loses its average: $Q^\pi(s, a) = r(s, a) + \gamma \sum_{s'} P(s' | s, a) V^{\pi_{s'}}(s')$. The two bounds repeat word for word, without the max:

$$Q^*(s, a) = r(s, a) + \gamma \sum_{s'} P(s' | s, a) V^*(s'). \quad (2)$$

Comparing with (1), $V^*(s) = \max_a Q^*(s, a)$; one into the other,

$$Q^*(s, a) = r(s, a) + \gamma \sum_{s'} P(s' | s, a) \max_{a'} Q^*(s', a').$$

Against (Q'_π) , one change: at s' , the average $\sum_{a'} \pi(a' | s')$ became a max over a' .

Proof of the Theorem: the greedy table is optimal. At every s let $\pi_g(s)$ be an action attaining the max in (1) ($\mathcal{A}$ is finite, so one exists): π_g is a deterministic stationary table.

- $V^*(s) = r(s, \pi_g(s)) + \gamma \sum_{s'} P(s' | s, \pi_g(s)) V^*(s')$ for every s : equation (1), its max attained by $\pi_g(s)$.
- $V^* = r^{\pi_g} + \gamma P^{\pi_g} V^*$: V^* solves (E_{π_g}) , the Bellman equation of the table π_g . That equation has exactly one solution, V^{π_g} ($I - \gamma P^{\pi_g}$ invertible). So $V^{\pi_g} = V^*$.
- $Q^{\pi_g}(s, a) = r(s, a) + \gamma \sum_{s'} P(s' | s, a) V^{\pi_g}(s') = r(s, a) + \gamma \sum_{s'} P(s' | s, a) V^*(s') = Q^*(s, a)$: (Q_{π_g}) , then equation (2).

One deterministic stationary table attains every sup in the definition: $\pi^* = \pi_g$, and sup = max from here on. The argument ran definition, equation, theorem; the derivation used no theorem. Uniqueness entered once, for the linear equation (E_{π_g}) of the fixed table π_g ; that the optimality equation itself has only one solution is a separate fact, stated below and proved with value iteration.

Hypothesis	Where the argument uses it	Without it
finite $\mathcal{A}$	the argmax defining π_g exists	one state, actions $a \in (0, 1)$ paying a : every rule is worth less than $1/(1 - \gamma)$ and some come within any distance of it; the sup is not attained, no optimal rule
finite $\mathcal{S}, \gamma < 1$	$I - \gamma P^{\pi_g}$ invertible, so (E_{π_g}) has one solution; every value finite	the machine at $\gamma = 1$: “repair when broken” earns $\frac{26}{7}$ a week forever, $V^* = \infty$, the equation has no solution
infinite horizon	($*$): from s' at time 1 the problem is the one from s' at time 0, so the continuation is worth $V^{\pi_{s'}}(s')$ and the same V^* bounds it	a fixed horizon H : an optimal rule exists and is deterministic, but it watches the clock (Lecture 2’s shop orders nothing in its last week)

Where the theorem needs its hypotheses. It is for the discounted, infinite-horizon, finite problem, the course’s default.

The machine’s optimality equation. Three unknowns. Each row is the better of two branches, run or repair; the repair branch is the same number in every row:

$$\begin{aligned}
 V^*(G) &= \max \left\{ 8 + 0.9 \left(\frac{2}{3} V^*(G) + \frac{1}{3} V^*(W) \right), -10 + 0.9 V^*(G) \right\}, \\
 V^*(W) &= \max \left\{ 4 + 0.9 \left(\frac{2}{3} V^*(W) + \frac{1}{3} V^*(B) \right), -10 + 0.9 V^*(G) \right\}, \\
 V^*(B) &= \max \left\{ 0 + 0.9 V^*(B), -10 + 0.9 V^*(G) \right\}.
 \end{aligned}$$

Fix a branch in every row and the maxes disappear: (E_{π}) for that table, three linear equations. Eight ways to fix them, eight tables. The equation asks for the one choice whose linear solution also wins every max, and which branch wins depends on the numbers the branches produce. The proof above exhibits such a choice, π_g , the branches V^* wins; finding it without V^* in hand is the problem.

Example. Solve it by hand: guess, solve, verify.

Guess the branches (run, run, repair). *Solve*: the chain is a cycle, so the three linear equations close into one. $V(B) = -10 + 0.9 V(G)$; $V(W) = 4 + 0.6 V(W) + 0.3 V(B)$ gives $V(W) = 10 + 0.75 V(B)$; $V(G) = 8 + 0.6 V(G) + 0.3 V(W)$ gives $V(G) = 20 + 0.75 V(W)$. Substituting downward, $V(G) = 21.875 + 0.50625 V(G)$, so $V(G) = 3500/79 = 44.30$, $V(W) = 2560/79 = 32.41$, $V(B) = 2360/79 = 29.87$. *Verify* the three maxes. At B: run $0.9 \cdot 29.87 = 26.89$, repair $-10 + 0.9 \cdot 44.30 = 29.87$; repair wins, as guessed. At G: run $8 + 0.9(\frac{2}{3} \cdot 44.30 + \frac{1}{3} \cdot 32.41) = 8 + 36.30 = 44.30$, repair 29.87; run. At W: run $4 + 0.9(\frac{2}{3} \cdot 32.41 + \frac{1}{3} \cdot 29.87) = 4 + 28.41 = 32.41$, repair 29.87; run. Every guessed branch wins its row, so $(44.30, 32.41, 29.87)$ solves all three equations, and the equation's only solution (stated below, proved with value iteration) is V^* . A wrong guess fails the verification in some row; that failure is the optimality test below.

	(E_π) : the equation for V^π	Today: the optimality equation
Unknown	V^π ; one policy fixed	V^* ; no policy fixed
At the root	the average $\sum_a \pi(a s)[\dots]$	the max $\max_a[\dots]$
Kind	$ \mathcal{S} $ linear equations	$ \mathcal{S} $ equations, nonlinear because of the max
Solving it	$V^\pi = (I - \gamma P^\pi)^{-1} r^\pi$, or iteration	no closed form; iteration with the max inside (value iteration, next)
Solutions	exactly one for $\gamma < 1$ ($I - \gamma P^\pi$ invertible)	exactly one for $\gamma < 1$: stated today, proved with value iteration

The Bellman optimality operator. One iteration with a max inside:

$$(\mathcal{T}V)(s) := \max_a \left[r(s, a) + \gamma \sum_{s'} P(s' | s, a) V(s') \right], \quad V^* = \mathcal{T}V^*.$$

$\mathcal{T}^\pi$, one iteration of (E_π) 's right-hand side, has $\sum_a \pi(a | s)$ where $\mathcal{T}$ has $\max_a$. For every V and every π , $\mathcal{T}^\pi V \leq \mathcal{T}V$ entry by entry (an average is at most the largest term), with equality exactly when π puts all its probability on maximizing actions: π is *greedy with respect to V* . Uniqueness, stated: V^* is the only solution of $V = \mathcal{T}V$; the proof is the contraction argument of value iteration.

Why the max makes it hard. In the machine’s three equations, fixing a branch per row turned the maxes into the linear system (E_π) of one table. The equation asks for the one choice of branches, one of $|\mathcal{A}|^{|\mathcal{S}|}$ tables, whose linear solution also wins every max: a search over policies in disguise. Solving it head-on is “akin to an exhaustive search” (Sutton and Barto, 2018, Section 3.6). Two notations elsewhere: B for the operator, and a version acting on Q tables (Jiang, 2024).

Method	How it attacks $V^\star = \mathcal{T}V^\star$
value iteration (Lecture 5)	iterates $V \leftarrow \mathcal{T}V$ from any start: no branches guessed, the iterates converge to the fixed point
policy iteration	guess a table, solve its linear system, re-pick the winning branches, repeat: the hand solution above, organized
Q-learning	the lookahead $r + \gamma \max_{a'} Q(s', a')$ evaluated on observed transitions instead of on P : no model
deep Q-networks	the same lookahead as the regression target for a Q table too large to store

One equation, the rest of the course: every method solves it, exactly or approximately (Sutton and Barto, 2018, Section 3.6).

From Values to a Policy

Greedy extraction (Sutton and Barto, 2018, Section 3.6)

- Input: $V^\star$ and the model (P, r, γ) .
- For every $s \in \mathcal{S}$: for every $a \in \mathcal{A}$, $Q^\star(s, a) \leftarrow r(s, a) + \gamma \sum_{s'} P(s' | s, a) V^\star(s')$; then $\pi^\star(s) \leftarrow \operatorname{argmax}_a Q^\star(s, a)$, any maximizer at a tie.

$|\mathcal{S}| |\mathcal{A}|$ one-step lookaheads of $O(|\mathcal{S}|)$ each. With $Q^\star$ in hand instead, $\pi^\star(s) = \operatorname{argmax}_a Q^\star(s, a)$ is a lookup: no model needed. “Greedy” means one step ahead; $V^\star$ has already priced everything after the step. The table returned is the π_g of the proof on page 6: optimal, by the theorem.

Every optimal table is greedy. For an optimal deterministic π and any s ,

$$V^\star(s) = V^\pi(s) = Q^\pi(s, \pi(s)) \leq Q^\star(s, \pi(s)) \leq \max_a Q^\star(s, a) = V^\star(s),$$

equality throughout, so $\pi(s)$ attains the max. With the proof: the optimal tables are exactly the tables that pick a maximizer of $Q^\star(s, \cdot)$ at every state. Greedy extraction finds them all, one per way of breaking ties.

The machine.

s	$V^*(s)$	$Q^*(s, \text{run})$	$Q^*(s, \text{repair})$
G	44.30	44.30	29.87
W	32.41	32.41	29.87
B	29.87	26.89	29.87

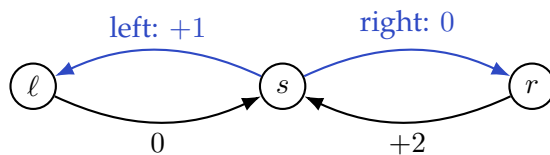
Q^* from V^* by one lookahead per pair. Bold: the maximizer, so $\pi^* = (\text{run}, \text{run}, \text{repair})$, “repair when broken”. The repair column is one number, $-10 + 0.9 \cdot 44.30$.

- At G and at W the machine still earns; a repair forfeits the week’s output and costs 10 to restore a machine that is good or nearly so.
- At B: $Q^*(B, \text{run}) = 0.9 \cdot 29.87 = 26.89$ is “run once more, then behave optimally”, i.e. repair next week. Waiting a week costs $29.87 - 26.89 = (1 - 0.9) \cdot 29.87 \approx 3$.
- $V^*(B) = 29.87 > 27.5 = V^\pi(G)$ under “never repair”: a broken machine, managed well, is worth more than a good one managed badly.

	Hold V^*	Hold Q^*
Size	$ \mathcal{S} $ numbers	$ \mathcal{S} \mathcal{A} $ numbers
Extracting π^*	one-step lookahead: needs P and r	table lookup: needs no model
Who computes it	planning with the model (value iteration, policy iteration)	planning; also learning from samples, with no model at all (Q-learning, later)

Ties. Two actions sharing the max at some state make π^* non-unique: either action, and any coin flip between them, is optimal; V^* and Q^* do not change. The machine at $\gamma \approx 0.62$ (the threshold below): $Q^*(B, \text{run}) = Q^*(B, \text{repair}) = 0$, and “never repair” and “repair when broken” are both optimal.

Example: two steps, one tie. Adapted from Sutton and Barto (2018, Exercise 3.22).



Only s has a choice. Labels are rewards; every transition is deterministic.

Which action at s ? Always left (π_L), always right (π_R):

$$V^{\pi_L}(s) = 1 + \gamma^2 V^{\pi_L}(s) \quad \implies \quad V^{\pi_L}(s) = \frac{1}{1 - \gamma^2},$$

$$V^{\pi_R}(s) = 2\gamma + \gamma^2 V^{\pi_R}(s) \quad \implies \quad V^{\pi_R}(s) = \frac{2\gamma}{1 - \gamma^2}.$$

γ	$0 \leq \gamma < \frac{1}{2}$	$\gamma = \frac{1}{2}$	$\frac{1}{2} < \gamma < 1$
Optimal choice at s	left	any mixture	right

Why any mixture? At $\gamma = \frac{1}{2}$, let $\pi_p(\text{left} \mid s) = p$, $0 \leq p \leq 1$:

$$V^{\pi_p}(s) = p \cdot 1 + (1 - p) \cdot 2\gamma + \gamma^2 V^{\pi_p}(s) = 1 + \frac{1}{4} V^{\pi_p}(s) \quad \implies \quad V^{\pi_p}(s) = \frac{4}{3} = V^*(s).$$

A deterministic optimal policy exists; stochastic policies can also be optimal.

A Test for Optimality

The opening comparison, 14.75 against 10 at W , was a verdict. Here is the rule it applied.

A policy is optimal if and only if its own values satisfy the optimality equation:

$$\pi \text{ optimal} \iff V^\pi(s) = \max_a Q^\pi(s, a) \text{ for every } s \iff V^\pi = \mathcal{T}V^\pi.$$

($\Rightarrow$) If π is optimal, $V^\pi = V^*$, which satisfies the equation. ($\Leftarrow$) If V^π satisfies it, V^π solves $V = \mathcal{T}V$; the solution is unique (stated above), so $V^\pi = V^*$. Cost: one policy evaluation, the linear solve of (E_π) , and one lookahead per state. No other policy is evaluated.

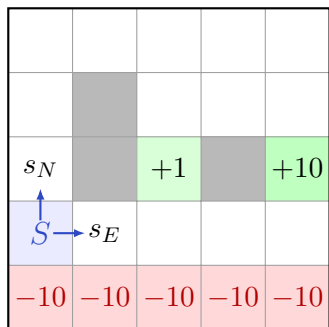
- “Never repair”: $V^\pi = (27.5, 10, 0)$, repair column 14.75. At W , $\max_a Q^\pi(W, a) = 14.75 > 10 = V^\pi(W)$. Fails: not optimal. One number was enough.
- “Repair when broken”: $V^\pi = (44.30, 32.41, 29.87)$; the hand solution above verified all three rows. Passes: optimal. None of the other seven rows was needed.
- “Repair when worn”: $V^\pi = (38.46, 24.62, 24.62)$. At W , run pays $4 + 0.9(\frac{2}{3} \cdot 24.62 + \frac{1}{3} \cdot 24.62) = 4 + 0.9 \cdot 24.62 = 26.15 > 24.62$: running a worn machine one more week beats repairing it now. Fails.

Example. The machine at $\gamma = 0.5$: is “never repair” optimal?

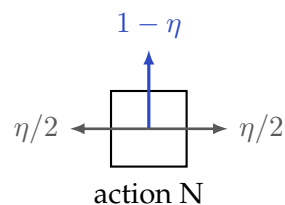
Evaluate it: $V^\pi(B) = 0$; $V^\pi(W) = 4 + 0.5 \cdot \frac{2}{3} V^\pi(W)$ gives 6; $V^\pi(G) = 8 + 0.5(\frac{2}{3} V^\pi(G) + \frac{1}{3} \cdot 6)$ gives 13.5. The repair column is $-10 + 0.5 \cdot 13.5 = -3.25$ in every row, below 13.5, 6, and 0. Passes: at $\gamma = 0.5$ “never repair” is optimal. The switch happens where $\gamma V^\pi(G) = 10$, the discounted value of a good machine just paying the repair bill: $\gamma \approx 0.62$.

Example: which exit, which route? Adapted from Abbeel (2012, Exercise 1).

- Gridworld variant: start at S ; actions N, E, S, W; walls and edges cause bumps.
- Noise η : intended move with probability $1 - \eta$, each sideways move with $\eta/2$.
- Pay +1, +10, or -10 on entering a colored exit; all other rewards 0. Exits are absorbing, with reward 0 afterwards.



Bottom row: cliff. Gray: walls.



Where one action can lead.

Predict: near or far? Along the cliff or away from it?

γ	η	Favored exit	$\pi^*(S)$
0.1	0	near (+1)	E
0.99	0	far (+10)	E
0.1	0.5	near (+1)	N
0.99	0.5	far (+10)	N

E: along the cliff. N: away from it.

With noise, the eventual exit is random.

No noise: shortest paths take 3 moves to +1, 5 to +10.

$$G_{\text{near}} = \gamma^2, \quad G_{\text{far}} = 10\gamma^4, \quad G_{\text{far}} > G_{\text{near}} \iff \gamma > \frac{1}{\sqrt{10}}.$$

Check at S : east can slip south into the cliff; north cannot.

$$Q^*(S, E) = (1 - \eta)\gamma V^*(s_E) + \frac{\eta}{2}\gamma V^*(s_N) - 10\frac{\eta}{2},$$

$$Q^*(S, N) = (1 - \eta)\gamma V^*(s_N) + \frac{\eta}{2}\gamma V^*(s_E) + \frac{\eta}{2}\gamma V^*(S).$$

- γ changes the price of delay; η changes the chance of falling.
- Same objective: expected discounted return. Changing P can change π^* .

Failure Mode: Greedy With Respect to the Wrong Values

Greedy extraction is a promise about V^* exactly. Fed other values, it returns a policy with no guarantee.

Values behind the lookahead	Greedy policy	Its value (G, W, B)
$V \equiv 0$: the immediate reward alone	never repair	(27.5, 10, 0)
V^π for π = never repair, (27.5, 10, 0)	repair when worn	(38.46, 24.62, 24.62)
$V^* = (44.30, 32.41, 29.87)$	repair when broken	(44.30, 32.41, 29.87)

- Row 1 is the myopic policy, greedy with respect to r : it is π^* for $\gamma = 0$ and for no other reason. Lecture 1's greedy traveler paid 7 for a route worth 5.
- Row 2 is better than its parent in every state and still not optimal. V^π prices "deviate once, then follow π "; the extracted policy deviates every time it returns to W or B, which V^π never priced. That the step improves anyway is the *policy improvement theorem*, proved later; repeating it is policy iteration.
- A V close to V^* but not equal to it (an iteration stopped early) can still produce a suboptimal greedy policy; how far from optimal is bounded by the error in V , settled with value iteration's stopping rule.

"Greedy" is relative. Always say with respect to what: greedy with respect to V^* is optimal; with respect to V^π , an improvement step; with respect to r , myopic. Lecture 1 said the same on the map: greedy with respect to V^* is optimal, greedy with respect to the immediate cost is not.

Summary

Lecture 4 on one board

- $V^*(s) = \sup_{\pi} V^{\pi}(s)$, $Q^*(s, a) = \sup_{\pi} Q^{\pi}(s, a)$ over all rules, per state; π^* is optimal if $V^{\pi^*} = V^*$. Theorem (proved from the equation): finite MDP, $\gamma < 1$: the table greedy w.r.t. V^* attains every sup at once, so $\sup = \max$; V^* and Q^* are unique, π^* need not be. Machine: $V^* = (44.30, 32.41, 29.87)$, $\pi^* = \text{"repair when broken"}$, the top row of eight.
- Bellman optimality: $V^*(s) = \max_a [r(s, a) + \gamma \sum_{s'} P(s' | s, a) V^*(s')]$; $Q^*(s, a) = r(s, a) + \gamma \sum_{s'} P(s' | s, a) \max_{a'} Q^*(s', a')$; $V^*(s) = \max_a Q^*(s, a)$. Derived from the definition: every rule splits at its first step into an action and a continuation rule (*); no rule beats one lookahead against V^* (upper bound); " a now, then the best rule from wherever you land" is a rule (lower bound). Then the theorem in three lines: V^* solves

the greedy table's Bellman equation (E_{π_g}), whose only solution ($I - \gamma P^{\pi_g}$ invertible) is that table's value. Machine: three equations, each the better of two branches; guess (run, run, repair), solve the cycle, verify $\max\{0.9 \cdot 29.87, -10 + 0.9 \cdot 44.30\} = 29.87$ at B and the other two rows.

- Operator: $(\mathcal{T}V)(s) = \max_a [\dots]$, $V^* = \mathcal{T}V^*$; $\mathcal{T}^\pi V \leq \mathcal{T}V$, with equality iff π is greedy w.r.t. V . Nonlinear: no linear solve. Value iteration next.
- Greedy extraction: $\pi^*(s) = \arg\max_a Q^*(s, a)$; from V^* one lookahead per pair (needs the model), from Q^* a lookup (needs none). The table returned is the proof's π_g ; every optimal table is greedy w.r.t. V^* .
- Test: π optimal iff $V^\pi(s) = \max_a Q^\pi(s, a)$ for all s . “Never repair” fails at W ($14.75 > 10$); “repair when broken” passes everywhere; at $\gamma = 0.5$ “never repair” passes: the optimal policy depends on γ .
- Failure mode: greedy w.r.t. r (myopic, 27.5) or w.r.t. V^π (38.46, better, not best) is not optimal; only exact V^* certifies (44.30).
- Next: computing V^* without guessing the policy, by iteration with the max inside (value iteration).

References

- Pieter Abbeel. Markov decision processes and exact solution methods. CS 287 course materials, University of California, Berkeley, 2012. URL <https://people.eecs.berkeley.edu/~pabbeel/cs287-fa12/slides/mdps-exact-methods.pdf>. Exercise 1: effect of discount and noise, pp. 16–20.
- Alekh Agarwal, Nan Jiang, Sham M. Kakade, and Wen Sun. Reinforcement learning: Theory and algorithms. Book draft, 2022. URL <https://rltheorybook.github.io/>.
- Richard Bellman. *Dynamic Programming*. Princeton University Press, 1957.
- Emma Brunskill. CS 234: Reinforcement learning. Stanford University course materials, 2026. URL <https://web.stanford.edu/class/cs234/>. Winter 2026 offering.
- Nan Jiang. MDP preliminaries. Lecture notes, University of Illinois Urbana-Champaign, 2024. URL <https://nanjiang.cs.illinois.edu/files/cs542f22/note1.pdf>. Dated October 5, 2019; updated September 27, 2024. Accessed 2026-09-13.
- Nan Jiang. CS 443: Reinforcement learning. University of Illinois Urbana-Champaign course materials, 2026. URL <https://nanjiang.cs.illinois.edu/cs443/>. Spring 2026 offering.
- Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994.
- Wen Sun. CS 4/5789: Introduction to reinforcement learning. Cornell University course

materials, 2025. URL https://wensun.github.io/CS4789_spring_2025.html. Spring 2025 offering.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2 edition, 2018. URL <http://incompleteideas.net/book/the-book-2nd.html>.