Markov Decision Processes and Bellman Equations

Emo Todorov

Applied Mathematics and Computer Science & Engineering

University of Washington

Winter 2014

Markov Decision Processes (MDPs)

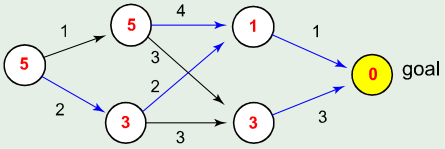
Notation and terminology:

$x \in \mathcal{X}$	state of the Markov process
$u \in \mathcal{U}(x)$	action/control in state x
$p(x'\|x, u)$	control-dependent transition probability distribution
$\ell(x, u) \geq 0$	immediate cost for choosing control u in state x
$q_{\mathcal{T}}(x) \geq 0$	(optional) scalar cost at terminal states $x \in \mathcal{T}$
$\pi(x) \in \mathcal{U}(x)$	control law/policy: mapping from states to controls
$v^{\pi}(x) \geq 0$	value/cost-to-go function: cumulative cost for starting at state x and acting according to π thereafter
$\pi^*(x)\,,\ v^*(x)$	optimal control law and corresponding value function

Different definitions of "cumulative cost" lead to different problem formulations.

Intuitions behind Dynamic Programming

Example (shortest paths)



- The optimal cost-to-go equals the immediate cost (for the optimal action at the current state) plus the optimal cost-to-go at the resulting next state.
- If the optimal cost-to-go is known, the optimal actions can be computed by greedy optimization, without explicitly considering future costs. This is because the optimal cost-to-go reflects all future costs.

First exit formulation

Each trajectory terminates when a terminal/goal state $x \in \mathcal{T}$ is first reached:

$$v^{\pi}(x) = E^{x_0=x}_{x_{k+1}\sim p(\cdot|x_k,\pi(x_k))}\left[q_{\mathcal{T}}\left(x_{t_{\text{first}}}\right) + \sum_{k=0}^{t_{\text{first}}-1} \ell\left(x_k, \pi\left(x_k\right)\right)\right]$$

At terminal states we have $v^{\pi}(x) = q_{\mathcal{T}}(x)$.

Definition (Hamiltonian)

$$H\left[x,u,v\left(\cdot\right)\right] \triangleq \ell\left(x,u\right) + E_{x'\sim p(\cdot|x,u)} v\left(x'\right)$$

Theorem (Bellman equations)

policy-specific cost-to-go: $v^{\pi}(x) = H\left[x, \pi(x), v^{\pi}(\cdot)\right]$

optimal cost-to-go: $v^{*}(x) = \min_{u\in\mathcal{U}(x)} H\left[x,u,v^{*}(\cdot)\right]$

optimal policy: $\pi^{*}(x) = \arg\min_{u\in\mathcal{U}(x)} H\left[x,u,v^{*}(\cdot)\right]$

Finite horizon formulation

This is a special case of the first-exit formulation. All relevant quantities are now indexed by time. All trajectories end at $t = N$:

$$v_t^{\pi}(x) = E_{x_{k+1} \sim p(\cdot|x_k, \pi_k(x_k))}^{x_t = x} \left[q_{\mathcal{T}}(x_N) + \sum_{k=t}^{N-1} \ell(x_k, \pi_k(x_k)) \right]$$

Unlike the general first-exit case (which is complicated when the transition graph contains loops) here the optimal cost-to-go can be found with a single backward pass through time.

Theorem (Bellman equations)

policy-specific cost-to-go: $v_t^{\pi}(x) = H\left[x, \pi_t(x), v_{t+1}^{\pi}(\cdot)\right]$

optimal cost-to-go: $v_t^{*}(x) = \min_{u \in \mathcal{U}(x)} H\left[x, u, v_{t+1}^{*}(\cdot)\right]$

optimal policy: $\pi_t^{*}(x) = \arg\min_{u \in \mathcal{U}(x)} H\left[x, u, v_{t+1}^{*}(\cdot)\right]$

Infinite horizon discounted cost formulation

The trajectory continues forever, but future costs are exponentially discounted (with $\alpha < 1$) to ensure that the cost-to-go remains finite:

$$v^{\pi}(x) = E^{x_0 = x}_{x_{k+1} \sim p(\cdot | x_k, \pi_k(x_k))} \left[\sum_{k=0}^{\infty} \alpha^k \ell(x_k, \pi(x_k)) \right]$$

The Hamiltonian now becomes $H_{\alpha}[x, u, v(\cdot)] \triangleq \ell(x, u) + \alpha E_{x' \sim p(\cdot | x, u)} v(x')$

Theorem (Bellman equations)

policy-specific cost-to-go: $v^{\pi}(x) = H_{\alpha}[x, \pi(x), v^{\pi}(\cdot)]$

optimal cost-to-go: $v^{*}(x) = \min_{u \in \mathcal{U}(x)} H_{\alpha}[x, u, v^{*}(\cdot)]$

optimal control law: $\pi^{*}(x) = \arg\min_{u \in \mathcal{U}(x)} H_{\alpha}[x, u, v^{*}(\cdot)]$

Smaller α makes the problem easier to solve, but the resulting policy can be short-sighted. This is relevant in economics and psychology.

Infinite horizon discounted cost formulation

The trajectory continues forever, but future costs are exponentially discounted (with $\alpha < 1$) to ensure that the cost-to-go remains finite:

$$v^{\pi}(x) = E^{x_0=x}_{x_{k+1}\sim p(\cdot|x_k,\pi_k(x_k))}\left[\sum_{k=0}^{\infty}\alpha^k \ell(x_k,\pi(x_k))\right]$$

The Hamiltonian now becomes $H_{\alpha}[x,u,v(\cdot)] \triangleq \ell(x,u) + \alpha E_{x'\sim p(\cdot|x,u)} v(x')$

Theorem (Bellman equations)

policy-specific cost-to-go: $v^{\pi}(x) = H_{\alpha}[x,\pi(x),v^{\pi}(\cdot)]$

optimal cost-to-go: $v^{*}(x) = \min_{u\in\mathcal{U}(x)} H_{\alpha}[x,u,v^{*}(\cdot)]$

optimal control law: $\pi^{*}(x) = \arg\min_{u\in\mathcal{U}(x)} H_{\alpha}[x,u,v^{*}(\cdot)]$

Smaller α makes the problem easier to solve, but the resulting policy can be short-sighted. This is relevant in economics and psychology.

Every discounted cost problem can be converted to a first exit problem. This is done by scaling the transition probabilities by α, introducing a terminal state with zero cost, and setting all transition probabilities to that state to $1-\alpha$.

Infinite horizon average cost formulation

The trajectory continues forever and there is no discounting, thus the cost-to-go is infinite. Let $v_0^{\pi,N}(x)$ denote the cost-to-go at time 0 for a finite-horizon problem with N steps. Then the average cost is

$$c^{\pi} = \lim_{N\to\infty} \frac{1}{N} v_0^{\pi,N}(x)$$

Note that c^{π} does not depend on the initial state. The optimal control law (minimizing c) can be found using a "differential" cost-to-go $\widetilde{v}$, which loosely speaking is

$$\widetilde{v}^{\pi}(x) = v_0^{\pi,N}(x) - Nc^{\pi}$$

Theorem (Bellman equations)

policy-specific cost-to-go: $c^{\pi} + \widetilde{v}^{\pi}(x) = H\left[x, \pi(x), \widetilde{v}^{\pi}(\cdot)\right]$

optimal cost-to-go: $c^{*} + \widetilde{v}^{*}(x) = \min_{u\in\mathcal{U}(x)} H\left[x, u, \widetilde{v}^{*}(\cdot)\right]$

optimal policy: $\pi^{*}(x) = \arg\min_{u\in\mathcal{U}(x)} H\left[x, u, \widetilde{v}^{*}(\cdot)\right]$

Policy evaluation using linear algebra

Stacking all states into a vector and suppressing the superscript π, the cost-to-go $v^{\pi}(x)$ becomes $\mathbf{v}$, the immediate cost $\ell(x, \pi(x))$ becomes $\mathbf{r}$, and the transition probability distribution $p(x'|x, \pi(x))$ becomes P where rows correspond to x and columns to x'. Let $\mathcal{N}$ be the set of non-terminal states.

First exit

$$\begin{aligned}\mathbf{v}_{\mathcal{N}} &= \mathbf{r}_{\mathcal{N}} + P_{\mathcal{N}\mathcal{N}}\mathbf{v}_{\mathcal{N}} + P_{\mathcal{N}\mathcal{T}}\mathbf{q}_{\mathcal{T}} \\ \mathbf{v}_{\mathcal{N}} &= \left(I - P_{\mathcal{N}\mathcal{N}}\right)^{-1}\left(\mathbf{r}_{\mathcal{N}} + P_{\mathcal{N}\mathcal{T}}\mathbf{q}_{\mathcal{T}}\right)\end{aligned}$$

Finite horizon

$$\mathbf{v}_n = \mathbf{r}_n + P_n\mathbf{v}_{n+1}$$

Infinite horizon discounted cost

$$\begin{aligned}\mathbf{v} &= \mathbf{r} + \alpha P\mathbf{v} \\ \mathbf{v} &= \left(I - \alpha P\right)^{-1}\mathbf{r}\end{aligned}$$

Infinite horizon average cost

$$c\mathbf{1} + \widetilde{\mathbf{v}} = \mathbf{r} + P\widetilde{\mathbf{v}}$$

Here $\widetilde{\mathbf{v}}$ is defined up to an additive constant. Fixing $\mathbf{1}^{\mathsf{T}}\widetilde{\mathbf{v}} = 0$ yields

$$\begin{bmatrix} \widetilde{\mathbf{v}} \\ c \end{bmatrix} = \begin{bmatrix} I - P & \mathbf{1} \\ \mathbf{1}^{\mathsf{T}} & 0 \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{r} \\ 0 \end{bmatrix}$$

Policy evaluation using linear algebra

Stacking all states into a vector and suppressing the superscript π, the cost-to-go $v^{\pi}(x)$ becomes $\mathbf{v}$, the immediate cost $\ell(x, \pi(x))$ becomes $\mathbf{r}$, and the transition probability distribution $p(x'|x, \pi(x))$ becomes P where rows correspond to x and columns to x'. Let $\mathcal{N}$ be the set of non-terminal states.

First exit

$$\begin{aligned} \mathbf{v}_{\mathcal{N}} &= \mathbf{r}_{\mathcal{N}} + P_{\mathcal{N}\mathcal{N}}\mathbf{v}_{\mathcal{N}} + P_{\mathcal{N}\mathcal{T}}\mathbf{q}_{\mathcal{T}} \\ \mathbf{v}_{\mathcal{N}} &= (I - P_{\mathcal{N}\mathcal{N}})^{-1}(\mathbf{r}_{\mathcal{N}} + P_{\mathcal{N}\mathcal{T}}\mathbf{q}_{\mathcal{T}}) \end{aligned}$$

Finite horizon

$$\mathbf{v}_n = \mathbf{r}_n + P_n\mathbf{v}_{n+1}$$

Infinite horizon discounted cost

$$\begin{aligned} \mathbf{v} &= \mathbf{r} + \alpha P\mathbf{v} \\ \mathbf{v} &= (I - \alpha P)^{-1}\mathbf{r} \end{aligned}$$

Infinite horizon average cost

$$c\mathbf{1} + \widetilde{\mathbf{v}} = \mathbf{r} + P\widetilde{\mathbf{v}}$$

Here $\widetilde{\mathbf{v}}$ is defined up to an additive constant. Fixing $\mathbf{1}^{\mathsf{T}}\widetilde{\mathbf{v}} = 0$ yields

$$\begin{bmatrix} \widetilde{\mathbf{v}} \\ c \end{bmatrix} = \begin{bmatrix} I - P & \mathbf{1} \\ \mathbf{1}^{\mathsf{T}} & 0 \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{r} \\ 0 \end{bmatrix}$$

The optimal control problem is well-defined when the matrix inverse exists for some policy. (Perron-Frobenius theory)

Value iteration

Recall that the optimal cost-to-go/value always satisfies an equation whose right-hand side is of form $\min_u H[x, u, v(\cdot)]$. This minimization operator is called the *dynamic programming backup* operator:

$$T[v(\cdot)](x) \triangleq \min_{u \in \mathcal{U}(x)} \left\{ \ell(x,u) + \alpha \sum_{x'} p(x'|x,u)\, v(x') \right\}$$

with $\alpha = 1$ in non-discounted problems. Now v^* is the solution to

$$v = T[v]$$

The easiest way to solve such fixed-point equations is

Algorithm (value iteration)

$$v^{(n+1)} = T\left[v^{(n)}\right]$$

Initialization can be arbitrary. In first-exit problems $v^{(n)}(x \in \mathcal{T}) = q_{\mathcal{T}}(x)$ for all n. In infinite-horizon average-cost problems we subtract the mean of v after each iteration (after convergence this mean is the average cost).

Convergence of value iteration

The Bellman equation for v^* has a unique solution (corresponding to the optimal cost-to-go) and value iteration converges to it. In the first exit and average cost problems some additional assumptions are needed:

- **First exit**: the algorithm converges to the unique optimal solution if there exists a policy with non-zero probability of termination starting from every state, and every infinitely long trajectory has infinite cost.
- **Finite horizon**: the algorithm always converges in N steps to the unique optimal solution.
- **Infinite horizon discounted cost**: the algorithm always converges to the unique optimal solution, i.e. the unique fixed point of $v = T[v]$, because the operator T is a contraction mapping (see below).
- **Infinite horizon average cost**: the algorithm converges to the unique optimal solution if every state can be reached from every other state (in finite time with non-zero probability) for some policy – which may depend on the pair of states.

Contraction mapping in discounted problems

Theorem (properties of T)

monotonicity	$v(x) \le v'(x), \forall x \implies T[v(\cdot)](x) \le T[v'(\cdot)](x)$
additivity	$T[v(\cdot)+d](x) = T[v(\cdot)](x) + \alpha d$
α-contraction	$\max_x \lvert T[v(\cdot)](x) - T[v'(\cdot)](x)\rvert \le \alpha \max_x \lvert v(x) - v'(x)\rvert$

Proof.

[Proof of α-contraction] Let $d = \max_x \lvert v(x) - v'(x)\rvert$. Then

$$v(x) - d \le v'(x) \le v(x) + d, \forall x$$

Apply T to both sides and use monotonicity and additivity:

$$T[v(\cdot)](x) - \alpha d \le T\left[v'(\cdot)\right](x) \le T[v(\cdot)](x) + \alpha d, \forall x$$

Therefore $\lvert T[v(\cdot)](x) - T[v'(\cdot)](x)\rvert \le \alpha d, \forall x$ $\square$

Policy iteration

Given π and v^{π}, we can improve the policy by choosing the action for which the Hamiltonian (i.e. the immediate cost plus the cost-to-go under π at the resulting next state) is minimal. The improvement can be made synchronously for all states:

Algorithm (policy iteration)

$$\pi^{(n+1)}(x) = \arg\min_{u \in \mathcal{U}(x)} H\left[x, u, v^{\pi^{(n)}}(\cdot)\right]$$

The initialization $\pi^{(0)}$ can be arbitrary as long as the resulting $v^{\pi^{(0)}}$ is finite (which is not always the case in first-exit problems).

Policy iteration always converges in a finite number of steps. This is because the number of different policies is finite (the state and control spaces are finite), and the algorithm cannot cycle since each iteration is an improvement.

Linear programming approach

We can relax the Bellman equation

$$v(x) = \min_{u \in \mathcal{U}(x)} \left\{ \ell(x,u) + \alpha \sum_{x'} p(x'|x,u)\, v(x') \right\}, \forall x$$

by replacing it with the set of linear inequality constraints

$$v(x) \leq \ell(x,u) + \alpha \sum_{x'} p(x'|x,u)\, v(x'), \forall (x,u) \qquad \text{(constraints)}$$

Now v^* is the maximal (at each state) function which satisfies all constraints, thus it can be found as

Algorithm (linear programming)

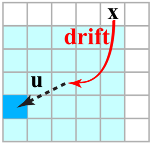
$$\max_{v(\cdot)} \sum_x v(x) \quad \text{s.t. (constraints)}$$

In the infinite-horizon average-cost formulation this becomes

$$\max_{c,\, \widetilde{v}(\cdot)} c$$

$$\text{s.t. } c + \widetilde{v}(x) \leq \ell(x,u) + \sum_{x'} p(x'|x,u)\, \widetilde{v}(x'), \forall (x,u) \text{ and } \sum_x \widetilde{v}(x) = 0$$

Example

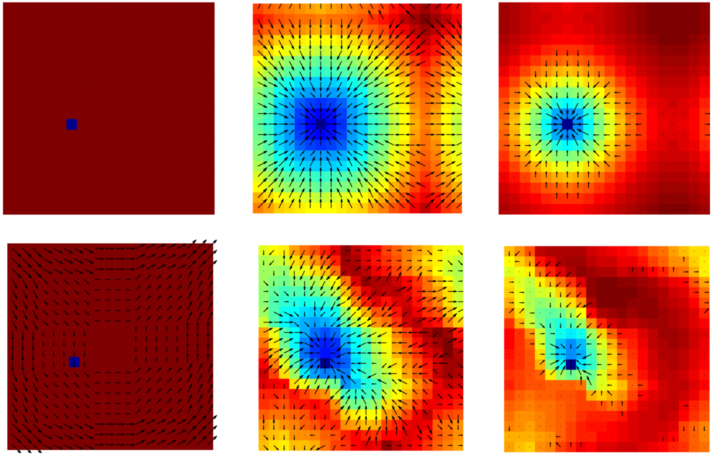


$\text{cost} = u^2 + q(x)$

drift and q(x)

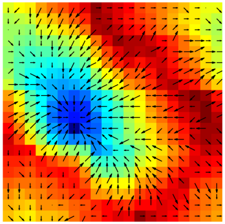
cost-to-go and policy

high noise

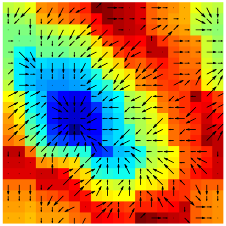


Example continued

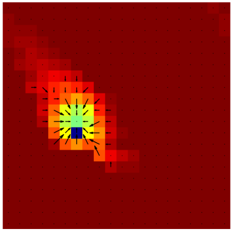
average



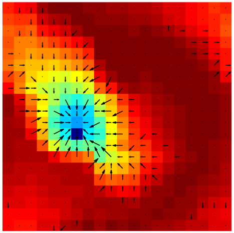
first exit



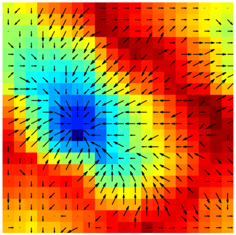
discounted, alpha = 0.6



alpha = 0.8



alpha = 0.99



Matlab: value iteration

```
[P, L, isgoal] = makeMDP;
[nx, nu] = size(L);
while 1,
    for iu = 1:nu
        H(:,iu) = L(:,iu) + alpha*P(:,:,iu)*v;
    end
    vold = v;
    [v, policy] = min(H,[],2);
    if average,
        c = mean(v);
        v = v - c;
    elseif firstexit,
        v(isgoal) = 0;
    end
    if max(abs(v-vold))<tol,
        break;
    end
end
```

Matlab: policy iteration

```
while 1,
    vold = v;
    for ix = 1:nx
        PP(ix,:)  = P(ix,:,policy(ix));
        LL(ix) = L(ix,policy(ix));
    end
    if discounted,
        v = (eye(nx)-alpha*PP)\LL;
    elseif average,
        tmp = [eye(nx)-PP, ones(nx,1); ones(1,nx), 0]\[LL; 0];
        v = tmp(1:nx);
        c = tmp(end);
    elseif firstexit,
        v(~isgoal) = (eye(sum(~isgoal)) - ...
                        PP(~isgoal,~isgoal)) \ L(~isgoal);
        v(isgoal) = 0;
    end
```

Matlab: policy iteration 2

```
    ...

    for iu = 1:nu
        H(:,iu) = L(:,iu) + alpha*P(:,:,iu)*v;
    end

    [v, policy] = min(H,[],2);
    if max(abs(v-vold))<tol,
        break;
    end
end
```

Matlab: linear programming

```
% min f'*v s.t.  A*v<=b, Aeq*v=beq
if discounted,
    f = -ones(nx,1);
    A = ones(nx*nu,nx);
    b = L(:);
elseif average,
    f = [zeros(nx,1); -1];
    A = ones(nx*nu,nx+1);
    b = L(:);
    Aeq = [ones(1,nx), 0];
    beq = 0;
elseif firstexit,
    nn = sum(~isgoal);
    f = -ones(nn,1);
    A = ones(nn*nu,nx);
    tmp = L(~isgoal,:);
    b = tmp(:);
end
```

Matlab: linear programming 2

```
...

% fill A matrix

ic = 1;
for iu = 1:nu
    for ix = 1:nx
        if ~(isgoal(ix) && firstexit),
            A(ic,1:nx) = -alpha*P(ix,:,iu);
            A(ic,ix) = 1-alpha*P(ix,ix,iu);
            ic = ic+1;
        end
    end
end
```

Matlab: linear programming 3

```
...

% run linprog

if discounted,
    v = linprog(f,A,b);
elseif average,
    tmp = linprog(f,A,b,Aeq,beq);
    v = tmp(1:nx);
    c = tmp(end);
elseif firstexit,
    A = A(:,~isgoal);
    tmp = linprog(f,A,b);
    v(~isgoal) = tmp;
    v(isgoal) = 0;
end
```