DYNAMIC PROGRAMMING

RECAP

PREVIOUSLY ON CMPUT 365

Compute v_π by iterative updates

$$v_i^{k+1} = \sum_a \pi(a \mid s_i)\left(r(s,a) + \gamma \sum_j p(s_i, a, s_j) v_j^k \right)$$

Can compute q_π the same way

DIRECTION

TODAY

Ways to use estimates of v_π to improve π

- Policy Improvement Theorem
- Policy Iteration
- Value Iteration

DYNAMIC PROGRAMMING

WHAT IS IT?

It is not the dynamic programming referred to in an algorithms courses

- The name comes from Bellman (1957)

Dynamic Programming is a class of algorithms that use value functions to organize and structure the search for π_*

Assume that a perfect model of the environment is known, e.g., $p(s', r \mid s, a)$ is known

- There is no agent interacting with an environment

IMPROVING π

USING VALUE ESTIMATES

Assume we know q_π. How can we choose actions to improve π?

IMPROVING π

USING VALUE ESTIMATES

Assume we know q_π. How can we choose actions to improve π?

$$\pi'(s) \in \max_a q_\pi(s, a)$$

π' is greedy with respect to q_π

Set action $A_t = \pi'(s)$ then take action according to π,

IMPROVING π

USING VALUE ESTIMATES

Assume we know q_π. How can we choose actions to improve π?

$$\pi'(s) \in \max_a q_\pi(s, a)$$

π' is greedy with respect to q_π

Set action $A_t = \pi'(s)$ then take action according to π,

It is not immediately clear that $v_{\pi'}(s) \geq v_\pi(s)$

$\max_a q_\pi(s, a) \overset{?}{=} v_{\pi'}(s)$ —usually not

POLICY IMPROVEMENT THEOREM

STATEMENT

For any policy π, if π' is a deterministic policy such that $\forall s \in \mathcal{S}$

$$q_\pi(s, \pi'(s)) \geq v_\pi(s),$$

then $\pi' \geq \pi$

POLICY IMPROVEMENT THEOREM

STATEMENT

For any policy π, if π' is a deterministic policy such that $\forall s \in \mathcal{S}$

$$q_\pi(s, \pi'(s)) \geq v_\pi(s),$$

then $\pi' \geq \pi$

Being greedy with respect to q_π leads to a better policy

POLICY IMPROVEMENT THEOREM

PROOF

$$\begin{aligned} v_\pi(s) &\leq q_\pi(s, \pi'(s)) \\ &\vdots \\ &= v_{\pi'}(s) \end{aligned}$$

POLICY IMPROVEMENT THEOREM

PROOF

$$
\begin{aligned}
v_\pi(s) \leq\ & q_\pi(s, \pi'(s)) \\
=\ & \mathbb{E}\left[R_{t+1} + \gamma v_\pi(S_{t+1}) \,|\, S_t = s, A_t = \pi'(s)\right] \\
\leq\ & \mathbb{E}\left[R_{t+1} + \gamma q_\pi(S_{t+1}, \pi'(S_{t+1})) \,|\, S_t = s, A_t = \pi'(s)\right] \\
=\ & \mathbb{E}\left[R_{t+1} + \gamma \mathbb{E}\left[R_{t+2} + \gamma v_\pi(S_{t+2}) \,|\, S_t = s, A_t = \pi'(S_t), A_{t+1} = \pi'(S_{t+1})\right] \,|\, S_t = s, A_t = \pi'(s)\right] \\
=\ & \mathbb{E}\left[R_{t+1} + \gamma R_{t+2} + \gamma^2 v_\pi(S_{t+2}) \,|\, S_t = s, A_t = \pi'(S_t), A_{t+1} = \pi'\right] \\
\leq\ & \mathbb{E}\left[R_{t+1} + \gamma R_{t+2} + \gamma^2 q_\pi(S_{t+2}, \pi'(S_{t+1})) \,|\, S_t = s, A_t = \pi'(S_t), A_{t+1} = \pi'\right] \\
\vdots\ & \\
\leq\ & \mathbb{E}\left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots \,\middle|\, S_t = s, A_t = \pi'(S_t), A_{t+1} = \pi'(S_{t+1}), A_{t+2} = \pi'(S_{t+2}), \ldots\right] \\
=\ & v_{\pi'}(s)
\end{aligned}
$$

SEARCHING FOR π_*

IDEA

Being greedy w.r.t q_π leads to a improved policy

Compute q_π —> compute π' as greedy on q_π —> compute $q_{\pi'}$ —> compute π'' —> …

POLICY ITERATION

ALGORITHM

$\pi^1 \leftarrow$ initialize π arbitrarily

for $k \in \{1,2,\ldots\}$

  $q_{\pi^k} \leftarrow$ compute_value$(\pi^k, \mathcal{S}, \mathcal{A}, p, r, \gamma)$

  $\forall s \in \mathcal{S},\ \pi^{k+1}(s) \leftarrow \arg\max_a q_{\pi^k}(s, a)$ // break ties deterministically

  if $\forall s \in \mathcal{S},\ \pi^{k+1}(s) = \pi^k(s)$ // terminate when policy stops changing

    return π^k // π^k is optimal

POLICY ITERATION

ALGORITHM

$\pi^1 \leftarrow$ initialize π arbitrarily

for $k \in \{1,2,\dots\}$

$q_{\pi^k} \leftarrow$ compute_value$(\pi^k, \mathcal{S}, \mathcal{A}, p, r, \gamma)$ Could take an infinite amount of time

$\forall s \in \mathcal{S},\ \pi^{k+1}(s) \leftarrow \arg\max_a q_{\pi^k}(s, a)$ // break ties deterministically

if $\forall s \in \mathcal{S},\ \pi^{k+1}(s) = \pi^k(s)$ // terminate when policy stops changing

return π^k // π^k is optimal

POLICY ITERATION

PROBLEMS

Computing the value function can take an infinite number of Bellman operator updates

New idea: perform K Bellman operator updates to estimate q_π

$\pi' \geq \pi$ — not guaranteed, but

$\lim_{k\to\infty} v_{\pi^k} \to v_{\pi_*}$, i.e., this process still finds the optimal policy eventually

For $K = 1$ this algorithm is called Value Iteration

VALUE ITERATION

ALGORITHM — VERSION 1

$v^0 \leftarrow$ initialize value estimates (usually $v_i^0 = 0$)

For $k \in \{1,2,3,\dots\}$

$$\forall i,\ \pi^k(s_i) \leftarrow \arg\max_a r(s_i, a) + \gamma \sum_j p(s_i, a, s_j) v_j^{k-1}$$

$$\forall i,\ v_i^k \leftarrow r(s_i, \pi^k(s_i)) + \gamma \sum_j p(s_i, \pi^k(s_i), s_j) v_j^{k-1}$$

If π^k OPTIMAL terminate

Repeated computation for π^k anbd v^k

VALUE ITERATION

ALGORITHM — VERSION 2

$v^0 \leftarrow$ initialize value estimates (usually $v_i^0 = 0$)

For $k \in \{1,2,3,\ldots\}$

$$\forall i,\ v_i^k \leftarrow \max_a r(s_i, a) + \gamma \sum_j p(s_i, a, s_j) v_j^{k-1}$$

Policy is implicitly defined based on v^k

If π^k OPTIMAL terminate

VALUE ITERATION

ALGORITHM — VERSION 2

$v^0 \leftarrow$ initialize value estimates (usually $v_i^0 = 0$)

For $k \in \{1,2,3,\dots\}$

$$\forall i,\ v_i^k \leftarrow \max_a r(s_i, a) + \gamma \sum_j p(s_i, a, s_j) v_j^{k-1}$$

If π^k OPTIMAL terminate

How do we know when to terminate?

VALUE ITERATION

TERMINATION

If $\forall i,\ v^{k+1} = v^k$ then $v^k = v_{\pi^k}$

VALUE ITERATION

TERMINATION

If $\forall i,\ v^{k+1} = v^k$ then $v^k = v_{\pi^k}$

Bellman Optimality Equation: $v_*(s) = \max_a r(s,a) + \gamma \sum_{s'} p(s,a,s')v_*(s')$

$\pi^k \in \Pi_*$, the optimal policy is reached

VALUE ITERATION

TERMINATION

If $\forall i,\ v^{k+1} = v^k$ then $v^k = v_{\pi^k}$

Bellman Optimality Equation: $v_*(s) = \max_a r(s,a) + \gamma \sum_{s'} p(s,a,s') v_*(s')$

$\pi^k \in \Pi_*$, the optimal policy is reached

For $k < \infty \ \ \|v^{k+1} - v^k\|_\infty > 0$

Takes an infinite number of updates for the value estimate to converge

VALUE ITERATION

TERMINATION

Stop when v^k is close to v_*, π^k is probably optimal, or at least good enough.

If $\|v^k - v_*\|_\infty \leq \epsilon$

Return π^k

How do we know $\|v^k - v_*\|_\infty$?

VALUE ITERATION

TERMINATION

If $\|v^{k+1} - v^k\|_\infty < \epsilon$

then $\|v_{\pi^k} - v_*\|_\infty \leq \dfrac{2\epsilon\gamma}{1-\gamma}$

For proof

Williams, Ronald J., and Leemon C. Baird. *Tight performance bounds on greedy policies based on imperfect value functions*. Tech. rep. NU-CCS-93-14, Northeastern University, College of Computer Science, Boston, MA, 1993.

VALUE ITERATION

ALGORITHM — VERSION 2

$v^0 \leftarrow$ initialize value estimates (usually $v_i^0 = 0$)

For $k \in \{1,2,3,\ldots\}$

$\forall i,\ v_i^k \leftarrow \max_a r(s_i, a) + \gamma \sum_j p(s_i, a, s_j) v_j^{k-1}$

If $\max_i |v_i^k - v_i^{k-1}| < \epsilon$

Return π^k

VALUE ITERATION

PROOF THAT $v^k \to v_*$

$$\lim_{k \to \infty} v^k \to v_*$$

Recall that the Bellman Operator was a contraction mapping

Thus, converged to v_π

Define $\mathcal{T} : \mathbb{R}^{|\mathcal{S}|} \to \mathbb{R}^{|\mathcal{S}|}$

$$\mathcal{T}(v)_i = \max_a r(s_i, a) + \gamma \sum_j p(s_i, a, s_j) v_j$$

Show $\mathcal{T}$ is a contraction mapping, then $v^k \to v_*$

VALUE ITERATION

PROOF THAT $v^k \to v_*$

Need to use the following property

Let $f\colon \mathscr{X} \to \mathbb{R}$ and $g : \mathscr{X} \to \mathbb{R}$ be functions on arbitrary sets $\mathscr{X}$

We have that:

$$\max_{x \in \mathscr{X}} f(x) - \max_{x \in \mathscr{X}} g(x) \leq \max_{x \in \mathscr{X}} |f(x) - g(x)|$$

VALUE ITERATION

CONTRACTION MAPPING

$$
\begin{aligned}
\|\mathcal{T}(v) - \mathcal{T}(v')\|_\infty &= \max_i |\mathcal{T}(v)_i - \mathcal{T}(v')_i| \\
&\vdots \\
&\leq \gamma \|v - v'|_\infty
\end{aligned}
$$

VALUE ITERATION

CONTRACTION MAPPING

$$
\begin{aligned}
\|\mathscr{T}(v)-\mathscr{T}(v')\|_\infty &= \max_i |\mathscr{T}(v)_i - \mathscr{T}(v')_i| \\
&= \max_i \left| \max_a \left(r(s,a) + \gamma \sum_j p(s_i,a,s_j)v_j \right) - \max_a \left(r(s,a) + \gamma \sum_j p(s_i,a,s_j)v'_j \right) \right| \\
&\leq \max_i \max_a \left| r(s,a) + \gamma \sum_j p(s_i,a,s_j)v_j - r(s,a) - \gamma \sum_j p(s_i,a,s_j)v'_j \right| \\
&= \max_i \max_a \left| \gamma \sum_j p(s_i,a,s_j)(v_j - v'_j) \right| \\
&= \max_i \max_a \gamma \sum_j p(s_i,a,s_j) \left| (v_j - v'_j) \right| \\
&\leq \max_i \max_a \gamma \sum_j p(s_i,a,s_j) \max_k \left| (v_k - v'_k) \right| \\
&= \max_i \max_a \gamma \max_k \left| (v_k - v'_k) \right| \sum_j p(s_i,a,s_j) \\
&= \max_i \max_a \gamma \max_k \left| (v_k - v'_k) \right| \\
&= \gamma \max_k \left| (v_k - v'_k) \right| \\
&= \gamma \|v - v'\|_\infty
\end{aligned}
$$

NEXT CLASS

WHAT YOU SHOULD DO

1. Programming assignment due Wednesday night

Wednesday: Exercises