Intro to DP

Wednesday, March 18, 2020 9:21 PM

Reinforcement Learning

Introduction to Dynamic Programming

What is Dynamic Programming ??

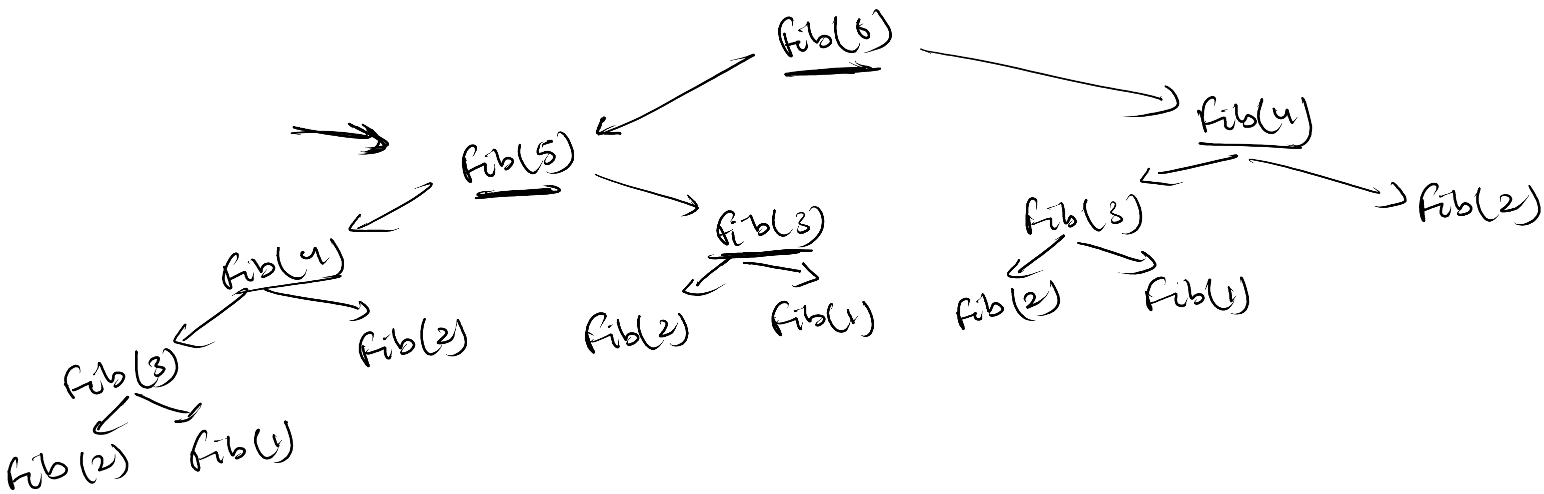
$K_{th} \longrightarrow fib(k)$

$fib(1) = 1$

$fib(2) = 1$

$fib(3) = 2$

$\underline{fib(n)} = \underline{fib(n-1)} + \underline{fib(n-2)}$



Dynamic programming for solving RL problems:

model based methods

Optimal policy.

perfect model of the environment

→ $p(s', r \mid s, a)$

Policy Evaluation

Wednesday, March 18, 2020 9:21 PM

Reinforcement Learning

Policy Evaluation

→ Policy Evaluation (Prediction).

→ Policy Improvement (Control).

$\pi \longrightarrow v_\pi$

Policy Evaluation using DP

Bellman euqation for v_π:

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$$

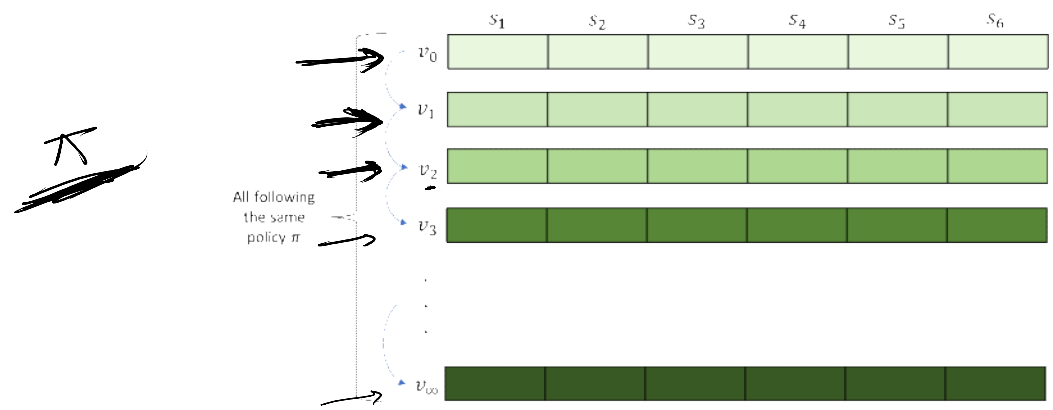
$\pi \longrightarrow v_\pi$

Update equation for policy evaluation:

$$v_{k+1}(s) = \sum_a \pi(a|s) \sum_{s',r} p(r,s'|s,a)[r + \gamma v_k(s')]$$

$v_0 \longrightarrow v_1 \longrightarrow v_2 \longrightarrow \cdots \longrightarrow v_k$.

$\lim_{k\to\infty} v_k = v_\pi$



$$\lim_{k\to\infty} v_k = v_\pi$$

Gridworld example

- State 0 and 15 are the terminal states.
- Agent is allowed to move UP, RIGHT, DOWN and LEFT.
- Each action deterministically cause the state transitions, except that actions which would take our agent off the grid, in such case the state remains unchanged.
- Reward of -1 on all transitions.
- Undiscounted and episodic.
- Policy to be evaluated is the equiprobable policy:

$$\pi(UP|s) = \pi(RIGHT|s) = \pi(DOWN|s) = \pi(LEFT|s) = 0.25 \ \forall s \in S$$

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

$\gamma = 1$

$$\frac{1}{4}\times 1\times[-1+(-2.0)] + \frac{1}{4}\times 1\times[-1+(-2.0)] + \frac{1}{4}\times 1\times[-1+(-1.7)] + \frac{1}{4}\times 1\times[-1+0]$$

$$= -2.4$$

v_0

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

v_1

0	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	-1
-1	-1	-1	0

v_2

0	-1.7	-2.0	-2.0
-1.7	-2.0	-2.0	-2.0
-2.0	-2.0	-2.0	-1.7
-2.0	-2.0	-1.7	0

v_3

0	-2.4	-2.9	-3.0
-2.4	-2.9	-3.0	-2.9
-2.9	-3.0	-2.9	-2.4
-3.0	-2.9	-2.4	0

v_∞

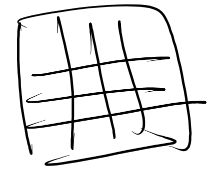
0	-14.	-20.	-22
-14	-18.	-20.	-20
-20	-20	-18	-14

v_{10}

0	-6.1	-8.4	-9.0
-6.1	-7.7	-8.4	-8.4
-8.4	-8.4	-7.7	-6.1

0	-14.	-20.	-22
-14	-18.	-20.	-20
-20	-20	-18	-14
-22	-20	-14	0

0	-6.1	-8.4	-9.0
-6.1	-7.7	-8.4	-8.4
-8.4	-8.4	-7.7	-6.1
-9.0	-8.4	-6.1	0



Policy Evaluation using DP (Algorithm)

Iterative Policy Evaluation, for estimating $V \approx v_\pi$

Input π, the policy to be evaluated
Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(terminal) = 0$

Loop:
  $\Delta \leftarrow 0$
  Loop for each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s', r \,|\, s, a)\big[r + \gamma V(s')\big]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Policy Improvement

Saturday, March 21, 2020 7:00 PM

Reinforcement Learning
Policy Improvement

$\pi \longrightarrow v_\pi$

$\pi' \geq \pi$

$v_{\pi'}(s) \geq v_\pi(s) \quad \forall s.$

's' $\quad \pi(s) = a \quad v_\pi(s).$

$q_\pi(s,a') = \sum_{s',r} p(s',r|s,a')[r + \gamma v_\pi(s')]$

$q_\pi(s,a') > v_\pi(s)$

$q_\pi(s,a) \rightarrow \pi(s).$

Policy Improvement Theorem:

Let π *and* π' *be any pair of deterministic policies s.t.,*

$$q_\pi(s,\pi'(s)) \geq v_\pi(s) \quad \forall \quad s \in S$$

Then $\pi' \geq \pi$*, that is,*

$$v_{\pi'}(s) \geq v_\pi(s) \quad \forall \quad s \in S$$

π, π'

π' $\quad \pi'(s) = a' \quad \pi(s) = a$

$\pi' \geq \pi$

Policy π' *is greedy wrt the value function of previous policy, i.e.,*

$$\pi'(s) = argmax_a \; q_\pi(s,a)$$
$$= argmax_a \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s)]$$

When the new policy π' is as good as the the previous policy π, then $v_\pi = v_{\pi'}$, hence

$$v_{\pi'}(s) = \max_a \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s)]$$

$$= \max_a \sum_{s',r} p(s',r|s,a)[r + \gamma v_{\pi'}(s)]$$

$$v_*(s) = \max_a \sum_{s',r} p(s',r|s,a)\left[r + \gamma v_*(s')\right]$$

$$v_{\pi'} = v_*$$

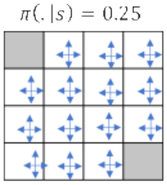
Gridworld example

- State 0 and 15 are the terminal states.
- Agent is allowed to move UP, RIGHT, DOWN and LEFT.
- Each action deterministically cause the state transitions, except that actions which would take our agent off the grid, in such case the state remains unchanged.
- Reward of -1 on all transitions.
- Undiscounted and episodic.
- Initial policy is an equiprobable policy:

$$\pi(UP|s) = \pi(RIGHT|s) = \pi(DOWN|s) = \pi(LEFT|s) = 0.25 \; \forall s \in S$$

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

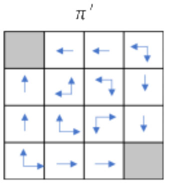
$\pi(.|s) = 0.25$



v_π

0	-14.	-20	-22
-14	-18.	-20.	-20
-20	-20	-18	-14
-22	-20	-14	0

π'



Reinforcement Learning
Policy Iteration

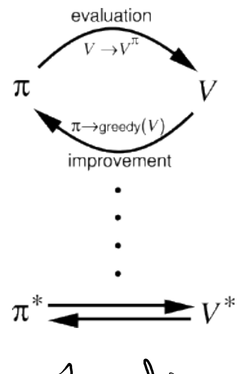
Policy Iteration

$$\pi_0 \xrightarrow{E} V_{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} V_{\pi_1} \xrightarrow{I} \pi_2 \rightarrow \cdots \rightarrow \pi_k \xrightarrow{E} V_{\pi_k} \xrightarrow{I} \pi_k$$

$$\pi_k = \pi_*$$

$$V_{\pi_k} = V_*$$

$$\pi_0 \xrightarrow{E} V_{\pi_0}$$
$$\pi_1 \xleftarrow{I} V_{\pi_0}$$
$$\pi_1 \xrightarrow{E} V_{\pi_1}$$
$$\pi_2 \xleftarrow{I} V_{\pi_1}$$
$$\vdots$$
$$\pi_* \longleftrightarrow V_*$$



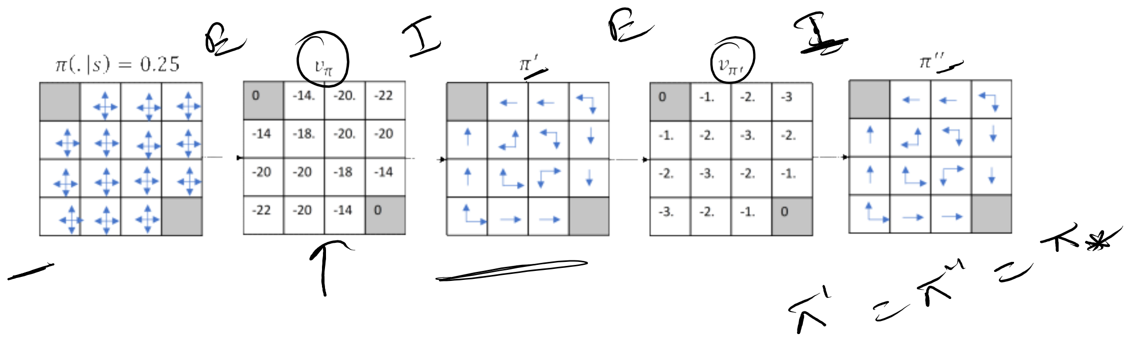
Generalized policy iteration.

Gridworld example

- State 0 and 15 are the terminal states.
- Agent is allowed to move UP, RIGHT, DOWN and LEFT.
- Each action deterministically cause the state transitions, except that actions which would take our agent off the grid, in su ch case the state remains unchanged.
- Reward of -1 on all transitions.
- Undiscounted and episodic.
- Initial policy is an equiprobable policy:

$$\pi(UP|s) = \pi(RIGHT|s) = \pi(DOWN|s) = \pi(LEFT|s) = 0.25\ \forall s \in S$$

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15



Policy Iteration Algorithm

Policy iteration (using iterative policy evaluation)

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Repeat
   $\quad \Delta \leftarrow 0$
   $\quad$ For each $s \in \mathcal{S}$:
   $\quad\quad v \leftarrow V(s)$
   $\quad\quad V(s) \leftarrow \sum_{s',r} p(s', r|s, \pi(s))\big[r + \gamma V(s')\big]$
   $\quad\quad \Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   $\quad$ *old-action* $\leftarrow \pi(s)$
   $\quad \pi(s) \leftarrow \operatorname{argmax}_a \sum_{s',r} p(s', r|s, a)\big[r + \gamma V(s')\big]$

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
      old-action $\leftarrow \pi(s)$
      $\pi(s) \leftarrow \operatorname{argmax}_a \sum_{s',r} p(s',r|s,a)\big[r + \gamma V(s')\big]$
      If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

Value Iteration

Monday, March 23, 2020 10:11 PM

Reinforcement Learning

Value Iteration

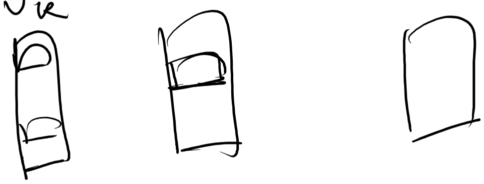
Value Iteration:

Value iteration update rule:

$$v_{k+1}(s) = \max_a \sum_{s',r} p(s',r|s,a)\big[r + \gamma v_k(s')\big] \quad \forall\ s \in S$$

$$v_*(s) = \max_a \sum_{s',r} p(s',r|s,a)\big[r + \gamma v_*(s')\big]$$

v_k v_{k+1}



Policy iteration (using iterative policy evaluation)

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Repeat
   - $\Delta \leftarrow 0$
   - For each $s \in \mathcal{S}$:
     - $v \leftarrow V(s)$
     - $V(s) \leftarrow \sum_{s',r} p(s',r|s,\pi(s))\big[r + \gamma V(s')\big]$
     - $\Delta \leftarrow \max(\Delta, |v - V(s)|)$

   until $\Delta < \theta$ (a small positive number)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   - *old-action* $\leftarrow \pi(s)$
   - $\pi(s) \leftarrow \arg\max_a \sum_{s',r} p(s',r|s,a)\big[r + \gamma V(s')\big]$
   - If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*

   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

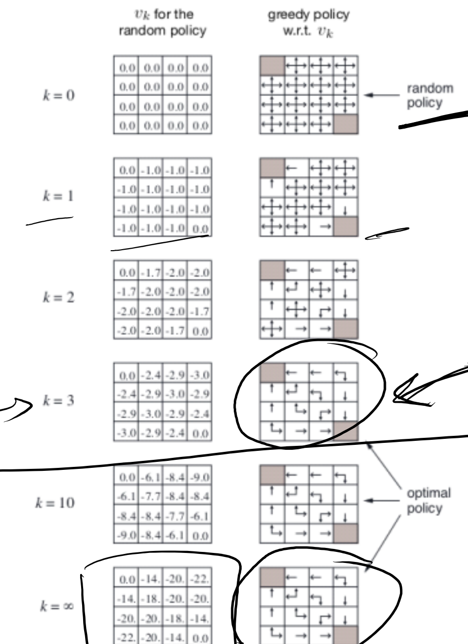
Gridworld example

- State 0 and 15 are the terminal states.

- Agent is allowed to move UP, RIGHT, DOWN and LEFT.
- Each action deterministically cause the state transitions, except that actions which would take our agent off the grid, in such case the state remains unchanged.
- Reward of -1 on all transitions.
- Undiscounted and episodic.
- Initial policy is an equiprobable policy:

$$\pi(UP|s) = \pi(RIGHT|s) = \pi(DOWN|s) = \pi(LEFT|s) = 0.25 \ \forall s \in S$$

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15



Value iteration algorithm

Value iteration

Initialize array V arbitrarily (e.g., $V(s) = 0$ for all $s \in \mathcal{S}^+$)

Repeat
  $\Delta \leftarrow 0$
  For each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \max_a \sum_{s',r} p(s', r|s, a)\left[r + \gamma V(s')\right]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$ (a small positive number)

Output a deterministic policy, $\pi \approx \pi_*$, such that
  $\pi(s) = \arg\max_a \sum_{s',r} p(s', r|s, a)\left[r + \gamma V(s')\right]$

Optimal value function.
greedy
Optimal policy.

$E \rightarrow I$

optimal policy.

$E \rightarrow I$

Optimal function $\longrightarrow$ Optimal policy.

Efficiency of DP

Wednesday, March 25, 2020 7:41 PM

Reinforcement Learning

Efficiency of DP based methods

$|S|$ $|A|$ $\left[|A|^{|S|} \right]$

P