Dynamic sequential or temporal component to the problem **Programming** optimizing a "program", i.e. a policy

Policy Evaluation

First, we look at *Policy Evaluation* which computes the state-value function v_π for an arbitrary policy π, this is also referred to as the *prediction problem*. Here we recall the definition of the state-value function,

$$\begin{aligned} v_\pi(s) &\doteq \mathbb{E}_\pi[G_t \mid S_t = s] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} \mid S_t = s] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s] \\ &= \sum_a \pi(a \mid s) \sum_{s',r} p\big(s', r \mid s, a\big)\big[r + \gamma v_\pi\big(s'\big)\big] \end{aligned}$$

As our approach to the solution is iterative, we start out with a sequence of approximations $v_0, v_1, v_3, \ldots$ each mapping $\mathcal{S}$ to $\mathbb{R}$. We obtain each approximation by using the bellman equation, we just introduced, as the update rule,

$$\begin{aligned} v_{k+1}(s) &\doteq \mathbb{E}_\pi[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s] \\ &= \sum_a \pi(a \mid s) \sum_{s',r} p\big(s', r \mid s, a\big)\big[r + \gamma v_k\big(s'\big)\big] \end{aligned}$$

for all $s \in \mathcal{S}$, where $v_k = v_\pi$ is fixed. Here, the sequence of $\{v_k\}$ can be shown to converge to v_π as $k \to \infty$ using the same conditions for the existence of v_π. This algorithm is called **iterative policy evaluation**. We update the values of these states by applying **expected updates**. The algorithm is defined as follows,

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

The computation of the state-value function helps us to find better policies. "*We know how good it is to follow the current policy from s - that is $v_\pi(s)$ -*)but would it be better or worse to change to the new policy?_" The way as described in Barto & Sutton is to consider selecting a in s and thereafter following the existing policy π, this approach yields,

$$\begin{aligned} q_\pi(s,a) &\doteq \mathbb{E}[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t = a] \\ &= \sum_{s',r} p\big(s', r \mid s, a\big)\big[r + \gamma v_\pi\big(s'\big)\big] \end{aligned}$$

The key idea here is that if there exists a value greater than v_π. That would mean there is exists a better policy to follow than π that selects a better action a based on state s. That this is true is a special case of a general result called the policy improvement theorem. Let π and π' be any pair of deterministic policies such that, for all $s \in \mathcal{S}$,

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

Hence, here the policy π' must be as good or better than, π. The proof of the policy improvement theorem is described as such,

$$\begin{aligned} v_\pi(s) &\leq q_\pi\big(s, \pi'(s)\big) \\ &= \mathbb{E}\big[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t = \pi'(s)\big] \\ &= \mathbb{E}_{\pi'}[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s] \\ &\leq \mathbb{E}_{\pi'}\big[R_{t+1} + \gamma q_\pi\big(S_{t+1}, \pi'(S_{t+1})\big) \mid S_t = s\big] \\ &= \mathbb{E}_{\pi'}\big[R_{t+1} + \gamma \mathbb{E}\big[R_{t+2} + \gamma v_\pi(S_{t+2}) \mid S_{t+1}, A_{t+1} = \pi'(S_{t+1})\big] \mid S_t = s\big] \\ &= \mathbb{E}_{\pi'}\big[R_{t+1} + \gamma R_{t+2} + \gamma^2 v_\pi(S_{t+2}) \mid S_t = s\big] \\ &\leq \mathbb{E}_{\pi'}\big[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 v_\pi(S_{t+3}) \mid S_t = s\big] \\ &\vdots \\ &\leq \mathbb{E}_{\pi'}\big[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \cdots \mid S_t = s\big] \\ &= v_{\pi'}(s). \end{aligned}$$

Now, to apply this we want to consider all states and actions, selecting at each state the action that appears best according to $q_\pi(s,a)$. This selection is done *greedy*, this is given by,

$$\begin{aligned} \pi'(s) &\doteq \underset{a}{\arg\max} q_\pi(s,a) \\ &= \underset{a}{\arg\max} \mathbb{E}[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s, A_t = a] \\ &= \underset{a}{\arg\max} \sum_{s',r} p\big(s', r \mid s, a\big)\big[r + \gamma v_\pi\big(s'\big)\big] \end{aligned}$$

Where on the $\arg\max_a$ operators the ties are broken arbitrarily. As the selection of the policy π' abides the Bellman Optimality equation, the found policy has to converge to the optimal policy. This approach is described as the *policy improvement* algorithm,

Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$

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

2. Policy Evaluation
   Loop:
   $\quad \Delta \leftarrow 0$
   $\quad$ Loop 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 determining the accuracy of estimation)

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

Which if combined with the policy evaluation algorithm yields *Policy Iteration*, as described above.

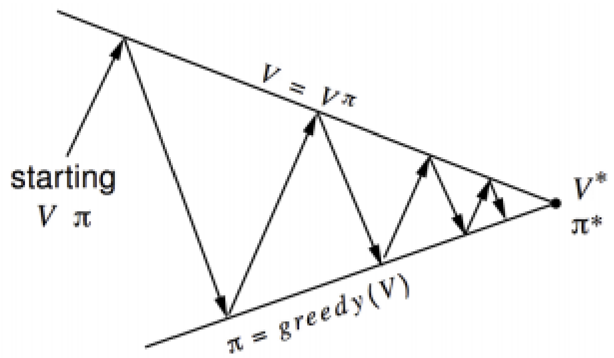
(Generalized) Policy Iteration

Policy iteration is used to for monotone improvement of the policies and value functions, and can be described as,

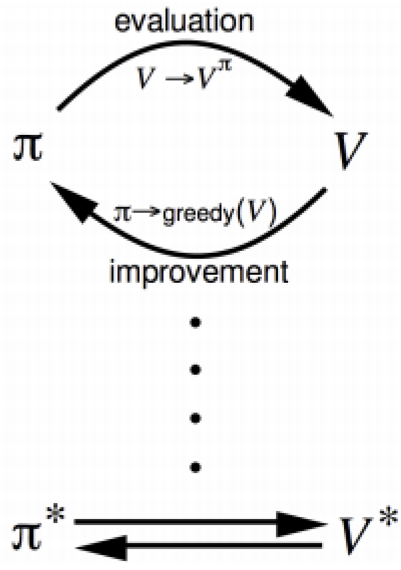
$$\pi_0 \xrightarrow{\text{E}} v_{\pi_0} \xrightarrow{\text{I}} \pi_1 \xrightarrow{\text{E}} v_{\pi_1} \xrightarrow{\text{I}} \pi_2 \xrightarrow{\text{E}} \cdots \xrightarrow{\text{I}} \pi_* \xrightarrow{\text{E}} v_*,$$

where $\xrightarrow{\text{E}}$ denotes a policy evaluation and $\xrightarrow{\text{I}}$ denotes a policy improvement. Because a finite MDP has only a finite number of policies, this process must converge to an optimal policy and optimal value function in a finite number of iterations.

This idea of Policy Iteration can be generalized further, hence *General Policy Iteration* (GPI). The idea of GPI does not only work in DP but also extends to other Reinforcement Learning techniques and is widely used. Now the process can be described as, the general idea of two interacting processes revolving around an approximate policy and an approximate value function.



Policy evaluation Estimate v_π
Any policy evaluation algorithm
Policy improvement Generate $\pi' \geq \pi$
Any policy improvement algorithm



Value Iteration

The Policy Iteration routine requires the execution of policy evaluation every iteration, which might introduce a large computational cost having to iterate over all states. Value Iteration presents an alternative of truncating the policy evaluation step without losing the convergence guarantees of policy Iteration. This adjustment proposed is stopping policy evaluation after one sweep, e.g. one update of each state. The update operation is defined as follows,

$$
\begin{aligned}
v_{k+1}(s) &\doteq \max_a \mathbb{E}[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s, A_t = a] \\
&= \max_a \sum_{s',r} p(s', r \mid s, a) \left[r + \gamma v_k(s')\right]
\end{aligned}
$$

for all $s \in \mathcal{S}$. This update can also be looked at as the Bellman Optimality equation.

Value Iteration, for estimating $\pi \approx \pi_*$

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 $s \in \mathcal{S}$:
| $\quad v \leftarrow V(s)$
| $\quad V(s) \leftarrow \max_a \sum_{s',r} p(s', r \mid s, a)\left[r + \gamma V(s')\right]$
| $\quad \Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

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

DP may not be practical for very large problems, but compared with other methods for solving MDPs, DP methods are actually quite efficient. If we ignore a few technical details, then the (worst case) time DP methods take to find an optimal policy is polynomial in the number of states and actions.

References

- Barto & Sutton, Reinforcement Learning: An introduction 2nd edition
- David Silver's Course on Reinforcement Learning