Implementação

- Algoritmo de iteração de valor

Algorithm 4.1: Value iteration algorithm

Initialization: The probability models $p(r|s,a)$ and $p(s'|s,a)$ for all (s,a) are known. Initial guess v_0.
Goal: Search for the optimal state value and an optimal policy for solving the Bellman optimality equation.

While v_k has not converged in the sense that $\|v_k - v_{k-1}\|$ is greater than a predefined small threshold, for the kth iteration, do

For every state $s \in \mathcal{S}$, do

For every action $a \in \mathcal{A}(s)$, do

q-value: $q_k(s,a) = \sum_r p(r|s,a)r + \gamma \sum_{s'} p(s'|s,a)v_k(s')$

Maximum action value: $a_k^*(s) = \arg\max_a q_k(s,a)$

Policy update: $\pi_{k+1}(a|s) = 1$ if $a = a_k^*$, and $\pi_{k+1}(a|s) = 0$ otherwise

Value update: $v_{k+1}(s) = \max_a q_k(s,a)$

Implementação

Algoritmo de iteração de política

Algorithm 4.2: Policy iteration algorithm

Initialization: The system model, $p(r|s,a)$ and $p(s'|s,a)$ for all (s,a), is known. Initial guess π_0.
Goal: Search for the optimal state value and an optimal policy.

While v_{π_k} has not converged, for the kth iteration, do
- *Policy evaluation:*
- Initialization: an arbitrary initial guess $v_{\pi_k}^{(0)}$
- While $v_{\pi_k}^{(j)}$ has not converged, for the jth iteration, do
  - For every state $s \in \mathcal{S}$, do
    - $v_{\pi_k}^{(j+1)}(s) = \sum_a \pi_k(a|s) \left[\sum_r p(r|s,a)r + \gamma \sum_{s'} p(s'|s,a) v_{\pi_k}^{(j)}(s')\right]$
- *Policy improvement:*
- For every state $s \in \mathcal{S}$, do
  - For every action $a \in \mathcal{A}$, do
    - $q_{\pi_k}(s,a) = \sum_r p(r|s,a)r + \gamma \sum_{s'} p(s'|s,a) v_{\pi_k}(s')$
  - $a_k^*(s) = \arg\max_a q_{\pi_k}(s,a)$
  - $\pi_{k+1}(a|s) = 1$ if $a = a_k^*$, and $\pi_{k+1}(a|s) = 0$ otherwise

Implementação

Algoritmo de iteração de política truncada

Algorithm 4.3: Truncated policy iteration algorithm

Initialization: The probability models $p(r|s,a)$ and $p(s'|s,a)$ for all (s,a) are known. Initial guess π_0.
Goal: Search for the optimal state value and an optimal policy.

While v_k has not converged, for the kth iteration, do
Policy evaluation:
Initialization: select the initial guess as $v_k^{(0)} = v_{k-1}$. The maximum number of iterations is set as j_{truncate}.
While $j < j_{\text{truncate}}$, do
For every state $s \in \mathcal{S}$, do
$v_k^{(j+1)}(s) = \sum_a \pi_k(a|s) \left[\sum_r p(r|s,a) r + \gamma \sum_{s'} p(s'|s,a) v_k^{(j)}(s') \right]$
Set $v_k = v_k^{(j_{\text{truncate}})}$
Policy improvement:
For every state $s \in \mathcal{S}$, do
For every action $a \in \mathcal{A}(s)$, do
$q_k(s,a) = \sum_r p(r|s,a) r + \gamma \sum_{s'} p(s'|s,a) v_k(s')$
$a_k^*(s) = \arg\max_a q_k(s,a)$
$\pi_{k+1}(a|s) = 1$ if $a = a_k^*$, and $\pi_{k+1}(a|s) = 0$ otherwise

$$\sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a) \left[r + \gamma v_k^{(j)}(s') \right]$$

Implementação

- Algoritmo: MC Básico

Algorithm 5.1: MC Basic (a model-free variant of policy iteration)

Initialization: Initial guess π_0.
Goal: Search for an optimal policy.

For the kth iteration ($k = 0, 1, 2, \dots$), do
    For every state $s \in \mathcal{S}$, do
        For every action $a \in \mathcal{A}(s)$, do
            Collect sufficiently many episodes starting from (s, a) by following π_k
            Policy evaluation:
            $q_{\pi_k}(s, a) \approx q_k(s, a) =$ the average return of all the episodes starting from (s, a)
        Policy improvement:
        $a_k^*(s) = \arg\max_a q_k(s, a)$
        $\pi_{k+1}(a|s) = 1$ if $a = a_k^*$, and $\pi_{k+1}(a|s) = 0$ otherwise

Implementação

- Algoritmo:

MC com inícios exploratórios

Algorithm 5.2: MC Exploring Starts (an efficient variant of MC Basic)

Initialization: Initial policy $\pi_0(a|s)$ and initial value $q(s,a)$ for all (s,a). Returns$(s,a) = 0$ and Num$(s,a) = 0$ for all (s,a).
Goal: Search for an optimal policy.

For each episode, do

Episode generation: Select a starting state-action pair (s_0, a_0) and ensure that all pairs can be possibly selected (this is the exploring-starts condition). Following the current policy, generate an episode of length T: $s_0, a_0, r_1, \ldots, s_{T-1}, a_{T-1}, r_T$.
Initialization for each episode: $g \leftarrow 0$
For each step of the episode, $t = T-1, T-2, \ldots, 0$, do

$g \leftarrow \gamma g + r_{t+1}$
Returns$(s_t, a_t) \leftarrow$ Returns$(s_t, a_t) + g$
Num$(s_t, a_t) \leftarrow$ Num$(s_t, a_t) + 1$
Policy evaluation:
$q(s_t, a_t) \leftarrow$ Returns(s_t, a_t)/Num(s_t, a_t)
Policy improvement:
$\pi(a|s_t) = 1$ if $a = \arg\max_a q(s_t, a)$ and $\pi(a|s_t) = 0$ otherwise