VI extensions

6. května 2019

B4M36PUI/BE4M36PUI — Planning for Artificial Intelligence

Outline

- Review of MDP concepts
- Value Iteration algorithm
- MDP solution
- Value function calculation

Review of last tutorial

Value function of a policy

Look at the following definition of a value function of a policy for inifnite-horizon MDP. It contains multiple mistakes, correct them on a piece of paper:

Def: Value function of a policy for infinite-horizon MDP

Assume infinite horizon MDP with $\gamma \in [0, 100]$. Then let Value function of a policy π for every state $s \in S$ be defined as

$$V^{\pi}(s) = \sum_{s' \in S} (s, \pi(s), s') R(s, \pi(s), s') + \gamma \pi(s')$$

Value function of a policy

Look at the following definition of a value function of a policy for inifnite-horizon MDP. It contains multiple mistakes, correct them on a piece of paper:

Def: Value function of a policy for infinite-horizon MDP

Assume infinite horizon MDP with $\gamma \in [0, 100]$. Then let Value function of a policy π for every state $s \in S$ be defined as

$$V^\pi(s) = \sum_{s' \in S} (s, \pi(s), s') R(s, \pi(s), s') + \gamma \pi(s')$$

$$\gamma \in [0, 1)$$

$$V^\pi(s) = \sum_{s' \in S} T(s, \pi(s), s')[R(s, \pi(s), s') + \gamma V^\pi(s')]$$

Value function of a policy

Look at the following definition of a value function of a policy for inifnite-horizon MDP. It contains multiple mistakes, correct them on a piece of paper:

Def: Value function of a policy for infinite-horizon MDP

Assume infinite horizon MDP with $\gamma \in [0, 100]$. Then let Value function of a policy π for every state $s \in S$ be defined as

$$V^\pi(s) = \sum_{s' \in S} (s, \pi(s), s') R(s, \pi(s), s') + \gamma \pi(s')$$

$$\gamma \in [0, 1)$$

$$V^\pi(s) = \sum_{s' \in S} T(s, \pi(s), s')[R(s, \pi(s), s') + \gamma V^\pi(s')]$$

Question: Difference to def. of an optimal value function?

Write down equations for finding a value function of a policy π. How would you solve these equations?

- S: S_0, S_1, S_2, S_3
- A: a_0, a_1, a_2

  $T(S_0, a_0, S_1) = 0.6$

  $T(S_0, a_0, S_2) = 0.4$
- T : $T(S_1, a_1, S_3) = 1$

  $T(S_2, a_2, S_3) = 0.7$

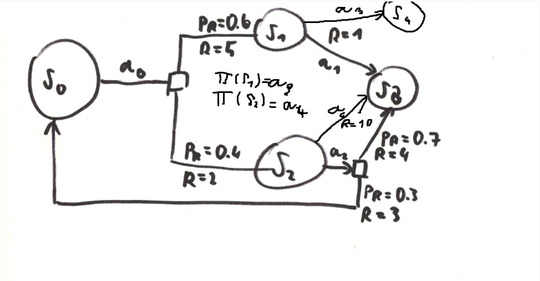
  $T(S_2, a_2, S_0) = 0.3$

  $R(S_0, a_0, S_1) = 5$

  $R(S_0, a_0, S_2) = 2$
- R : $R(S_1, a_1, S_3) = 1$

  $R(S_2, a_2, S_3) = 4$

  $R(S_2, a_2, S_0) = 3$



Value Iteration

value Iteration algorithm

Basic algorithm for finding solution of Bellman Equations iteratively.

1. initialize V_0 arbitrarily for each state, e.g to 0, set $n = 0$
2. Set $n = n + 1$.
3. Compute Bellman Backup, i.e. for each $s \in S$:
    3.1 $V_n(s) = \max_{a \in A} \sum_{s' \in S} T(s, a, s')[R(s, a, s') + \gamma V(s')]$
4. GOTO 2.

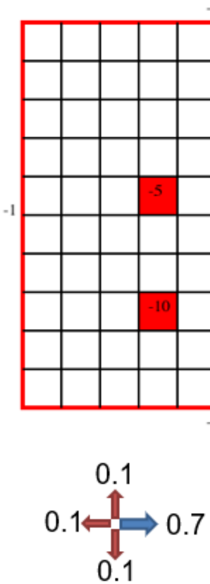
value Iteration algorithm

Basic algorithm for finding solution of Bellman Equations iteratively.

1. initialize V_0 arbitrarily for each state, e.g to 0, set $n = 0$
2. Set $n = n + 1$.
3. Compute Bellman Backup, i.e. for each $s \in S$:
    3.1 $V_n(s) = \max_{a \in A} \sum_{s' \in S} T(s, a, s')[R(s, a, s') + \gamma V(s')]$
4. GOTO 2.

Question: Does it converge? When do we stop?

VI example



Residual - definition

Def: Residual

Residual of value function V *from* V' *at state* $s \in S$ is defined by:

$$|V(s) - V'(s)|$$

Residual - definition

Def: Residual

Residual of value function V from V' at state $s \in S$ is defined by:

$$|V(s) - V'(s)|$$

Residual of value function V from V' is given by:

$$||V - V'||_\infty = \max_s |V(s) - V'(s)|$$

VI stopping criterion

Stopping criterion: When residual of consecutive value functions is below low value of ϵ:

$$||V_n - V_{n+1}|| < \epsilon$$

However, this does not imply ϵ distance of value of greedy policy from optimal value function.

Theorems exist of form:

$$V_n, V^* \text{as above} \rightarrow \forall s |V_n(s) - V^*(s)| < \epsilon \max\{N^*(s), N^{\pi^{V_n}}(s)\}$$

Value iteration properties

- Convergence: VI converges from any initialization (unlike PI)
- Termination: when resiudal is "small"
- Fixed point: V^* is a fixed point of Bellman operator
- Corollary: VI is monotonic

VI improvements

Asynchronous VI

Prioritized VI

- Prioritized sweeping

Partitioned VI

- Topological VI