COS402- Artificial Intelligence
Fall 2015

Lecture 17: MDP: Value Iteration and Policy Iteration

Outline

- **The Bellman equation and Bellman update**
- **Contraction**
- **Value iteration**
- **Policy iteration**

The Bellman equations for utilities

- **The relationship between the utility of a state and the utility of its neighbors**
  - $\mathbf{U}(\boldsymbol{s}) = \mathbf{R}(\boldsymbol{s}) + \boldsymbol{r}.\max_{\boldsymbol{a} \epsilon \boldsymbol{Actions}(\boldsymbol{s})} \sum_{\boldsymbol{s}'} \mathbf{P}(\mathbf{s}'|\mathbf{s},\mathbf{a})\, \boldsymbol{U}(\boldsymbol{S}')$
  - **Assuming the agent chooses the optimal action**
- **What is the best action in state (1,1)? in state (3,4)?**
- **How many Bellman equations do we have for this MDP?**
- **Can we solve these equations directly and efficiently?**

1	0.812	0.868	0.918	+1
2	0.762		0.660	-1
3	0.705	0.655	0.611	0.388
	1	2	3	4

Value iteration: Idea

- **Start with estimate $U_0 = 0$,**
- **Keep plugging in current estimate U_i to get new estimate U_{i+1};**
- **Repeat until little or no change in estimation.**

Value iteration: Algorithm

- **Initialize U_0 (S) = 0 for all S**
- **For I = 0,1,2, …**
    - **For all S, $\mathbf{U}_{i+1}(\mathbf{s}) = \mathbf{R}(\mathbf{s}) + \boldsymbol{r}.\max_{a \epsilon Actions(s)} \sum_{s'}$ P(s'|s,a) $U_i(S')$**
        - **"Bellman update"**
- **If $\max_{s} |\mathbf{U}_{i+1}(\mathbf{s})$-$\mathbf{U}_{i}(\mathbf{s})| < \epsilon$, stop and output $U_{i+1.}$**
- **For all S, Π*(s) = $\boldsymbol{arg}\max_{\mathbf{a}} \sum_{s'}$ P(s'|s,a) $U(S')$**

Value iteration: Does it work?

- **A contraction is a function of one argument. When applied to two different inputs in turn, the output values are getting "closer together".**
  - **A contraction has one fixed point.**
  - **Ex. "divided by 2" is a contraction. The fixed point is 0.**
- **Bellmen update is a contraction. Its fixed point it the vector/point of the true utilities of the states.**
- **The estimate of utility at each iteration is getting closer to the true utility.**

Policy iteration: Algorithm

- **Start with any policy Π_0,**
- **For i = 0,1,2, …**
  - **Evaluate: compute $U^{\Pi_i}(s)$**
  - **Greedify: $\Pi_{i+1}(s) = \arg\max_{a} \sum_{s'} P(s'|s,a) U^{\Pi_i}(S')$**
  - **Stop when $\Pi_{i+1} = \Pi_i$.**

Policy iteration: how to evaluate **Π** ?

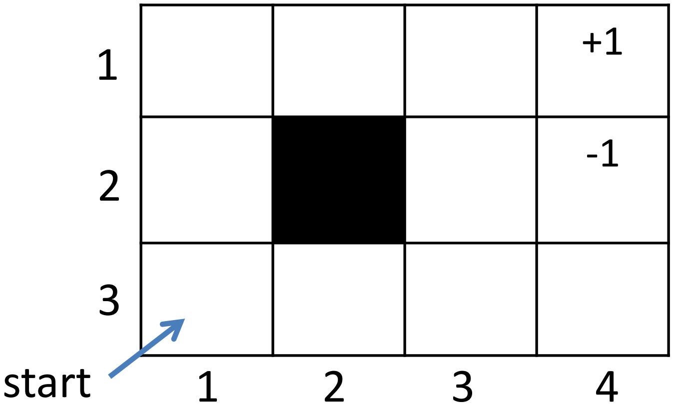
- **Iterative approach – simplified value iteration.**
  - **Like value iteration, except now action at state S is fixed to be Π(S).**
  - $U_{i+1}^{\Pi}(s) = R(s) + r.\sum_{s'} P(s'|\Pi(s),a)\, U_i^{\Pi}(S')$
- **Direct approach.**
  - $U^{\Pi}(s) = R(s) + r.\sum_{s'} P(s'|\Pi(s),a)\, U^{\Pi}(S')$
  - **A system of linear equations, can be solved directly in $O(n^3)$.**
  - **Efficient for small state spaces.**

Policy iteration: why does it work ?

- **Can prove (Policy improvement theorem)**
  - **$U^{\Pi_{i+1}}(s) \geq U^{\Pi_i}(s)$, with strict inequality for some s unless $\Pi_i = \Pi^*$**
- **Means policies getting better and better Π_{i+1}**
  - **Will never visit same policy Π twice**
  - **Will only terminate when reach Π***
- **#iterations <= #policies**
  - **In practice, no case found where more than O(n) iterations are needed.**
  - **Open question: does policy iteration converge in O(n)? (n is the number of that states in the MDP)**

POMDP: Example and definition

- **A robot in a grid**



- **MDP:**
  - **Initial state and states: hidden**
  - **Actions:**
  - **Transition model: P(s'|s,a)**
  - **Rewards: R(s)**
  - **Observation model: P(o|s)**

Review questions: true or false

1. **Value iteration is an algorithm for estimating the true utility of each state in a MDP.**

2. **The n (n is the number of states) Bellman equations for utility in a MDP can uniquely determine the true utilities of the states. These equations can be solved directly since they are exactly n variables and n equations.**

3. **Bellman update is a contraction. The fixed point is the vector/point of the true utilities of the states.**

4. **To evaluate a policy (compute $U^{\Pi}(s)$), we can write n Bellman equations with the actions fixed as Π(s). A simplified value iteration algorithm can be used to solve them since they can not be solved directly.**

Review questions: true or false(cnt'd)

5. **In Policy iteration, a policy will not be visited twice. Each iteration will lead to a new policy that is strictly better than the last one for at least one state.**
6. **The number of different policies is m^n (m is the average number of actions available for each state, and n is the number of states in a MDP). So policy iteration usually takes exponential time to run.**
7. **Policy iteration is guaranteed to terminate and find an optimal policy.**
8. **In POMDPs (partially observable MDPs), the agent does not know the state it is in. In stead of a transition model P(s'|s,a), it has an observation model P(o|s).**

Announcement & Reminder

- **W4 is due on Tuesday Nov. 24th**
  - **Turn in hard copy in class.**
- **P4 has been released and is due on Tuesday Dec. 1st**
  - **Upload files to CS dropbox by midnight.**