Lecture 18:

Q Learning

Q Learning

learning optimal policy $\pi^*(s)$ entails learning V^*, but this requires knowing $r()$ and $\delta()$

not typically known, so must learn an optimal policy for an unknown environment, thru exploration

Q-function: value of executing a followed by optimal policy

$$Q(s,a) = r(s,a) + \gamma V^*(\delta(s,a))$$

$$\pi^*(s) = \arg\max_a Q(s,a)$$

$$Q(s,a) = r(s,a) + \gamma \max_{a'} Q(\delta(s,a),a')$$

purely local decisions – base action choices on Q values for current state – will be optimal policy

Learning rule:

$$\hat{Q}(s,a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s',a')$$

Q learning algorithm

assumes deterministic rewards and actions

1. Initialize $\widehat{Q}(s,a) \leftarrow 0 \quad \forall s,a$
2. Observe current state s
3. Repeat forever
   - select action a and execute it
   - earn immediate reward r
   - observe new state s'
   - update $\widehat{Q}(s,a) \leftarrow r + \gamma \max_{a'} \widehat{Q}(s',a')$
   - update $s \leftarrow s'$

Q learning setup

training set consists of series of episodes – sequence of (state,action,reward) triples ending at absorbing state

for each executed action a resulting in a transition from state s_i to state s_j, the algorithm updates $\hat{Q}(s_i, a)$ using the learning rule

intuition for simple grid world, reward only entering goal state:

1. all $\hat{Q}(s, a)$ start at 0
2. first episode – only update $\hat{Q}(s, a)$ for transition leading to goal state
3. next episode – if go thru this next-to-last transition, will update $\hat{Q}(s, a)$ another step back
4. eventually propagate information from transitions with non-zero reward througout state-action space

Q learning convergence

under certain conditions, $\hat{Q}$ will converge to Q:

- deterministic actions and rewards
- bounded rewards
- every state-action pair must be visited infinitely often

proof: can show that error decreases monotonically:

$$|\hat{Q}_{n+1}(s,a) - Q(s,a)| \leq \gamma \max_{s',a'} |\hat{Q}_n(s',a') - Q(s',a')|$$

1. $\hat{Q}_{n+1}(s,a) \geq \hat{Q}_n(s,a)$
2. $0 \leq \hat{Q}_n(s,a) \leq Q(s,a)$

exploration/exploitation tradeoff – learn model faster vs. execute best action according to current model: can make policy select best action 95%, random action 5%