Markov Games
Ron Parr

CSCI 2951-F
Brown University

Alternating move, zero-sum, 2-player games

- Ordinary bellman equation

$$V(s) = \max_a R(s,a) + \gamma \sum_{s'} P(s'|s,a)V(s')$$

- Two player

$$V_{\max}(s) = \max_a R(s,a) + \gamma \sum_{s'} P(s'|s,a)V_{\min}(s')$$
$$V_{\min}(s) = \min_a R(s,a) + \gamma \sum_{s'} P(s'|s,a)V_{\max}(s')$$

V_{min} and V_{max}

- Vmin is, by definition, the negative of Vmax
- No need to store two separate value functions
- Can store just one and flip the sign based upon who is playing

Algorithms for zero-sum games

- Alternative move case is easy
- Minor generalizations of value iteration, policy iteration, Q-learning, etc. all work as expected
- Value function approximation works as in regular MDPs
- Used in:
  - TD-gammon
  - AlphaGo
- Not used in:
  - Atari Games (opponents are viewed as part of environment)

Why not treat opponent as part of environment?

- Want a strategy that is robust against all possible opponent actions
- When we maximizing and assume opponent is minimizing
  - Maximize our worst case results
  - If policy is optimal, then no opponent can force us to get less (in expectation) than values our computed for our value function
- If opponent is viewed as part of the environment
  - Implicitly assumes opponent behavior will not change in response to yours
  - If opponent policy does change:
    - Like learning in a non-stationary MDP
    - Learned policy can oscillate
    - Opponent can exploit your policy

Zero Sum Markov Games (simultaneous move)

- Combine MDPs with zero sum games
- Each state has a payoff matrix
- Joint action take by both players determines:
  - Immediate payoff
  - Distribution over next actions
- Question: Can we generalize MDP algorithms to this case?
- Answer: Yes!

Bellman equation for zero sum stochastic games

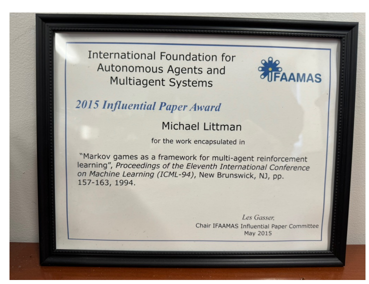
- Let Q(s,a,o) be the expected value (from player 1's perspective) when player 1 takes action a, player 2, takes action 0, and both players act optimally thereafter
- Then V(s) is the solution to the following linear program:

$$
\begin{array}{ll}
\text{Maximize:} & V(s) \\
\text{Subject to:} & \forall a \in \mathcal{A},\ \pi\ (s,a) \geq 0 \\
 & \sum_{a \in \mathcal{A}} \pi'(s,a) = 1 \\
 & \forall o \in \mathcal{O},\ V(s) \leq \sum_{a \in \mathcal{A}} Q(s,a,o)\pi\ (s,a)
\end{array}
$$

All of this replaces **max** in the Bellman equation

Application of modified Bellman equation

- Can we do value iteration, policy iteration, function approximation, Q-learning, etc. with this formulation?
- Yes! Notable example: minimax-Q
- But...
  - It's expensive
  - Updating every state requires solving a linear program



General Sum Stochastic Games

- Combine:
  - General sum games
  - MDPs
- Each state has:
  - Payoff matrix (not assumed to be zero sum)
  - Joint action determines immediate reward, distribution over next states
- Question: Can we use MDP techniques to find equilibria of general sum stochastic games?
- Answer: Not really ☹

Why MDP techniques fail for general sum stochastic games

- Recall the zero sum case:

$$\begin{aligned}
\text{Maximize:} \quad & V(s) \\
\text{Subject to:} \quad & \forall a \in \mathcal{A},\ \pi\ (s,a) \geq 0 \\
& \sum_{a \in \mathcal{A}} \pi^{'}(s,a) = 1 \\
& \forall o \in \mathcal{O},\ V(s) \leq \sum_{a \in \mathcal{A}} Q(s,a,o)\pi\ (s,a)
\end{aligned}$$

- Problems:
  - V(s) is not the result of a maximization in the general sum case
  - Our policy for state s should be an equilibrium policy, but which equilibrium?

Solution strategies

- A huge one-shot game where actions are policies? (Ew…, but ok…)
- What is a best response?
  - If opponent strategies are frozen, best response is solution to an MDP
  - Suggests numerous approaches such as double oracle, iterated best response, fictitious play, etc.
- Generalizations of minimax-Q -> Nash-Q
- RL approaches to general sum stochastic games typically converge only in special cases, not in general

General sum MG as a math program

$$\begin{aligned} \underset{\boldsymbol{\pi}, U}{\text{minimize}} \quad & \sum_{i \in \mathcal{I}} \sum_{s} \left(U^i(s) - Q^i(s, \boldsymbol{\pi}(s)) \right) \\ \text{subject to} \quad & U^i(s) \geq Q^i(s, a^i, \boldsymbol{\pi}^{-i}(s)) \text{ for all } i, s, a^i \\ & \sum_{a^i} \pi^i(a^i \mid s) = 1 \text{ for all } i, s \\ & \pi^i(a^i \mid s) \geq 0 \text{ for all } i, s, a^i \end{aligned}$$

where

$$Q^i(s, \boldsymbol{\pi}(s)) = R^i(s, \boldsymbol{\pi}(s)) + \gamma \sum_{s'} T(s' \mid s, \boldsymbol{\pi}(s)) U^i(s')$$

Non-linear terms highlighted

Summary

- Zero sum case is easy
  - Alternating move behaves just like and MDP; all algorithms generalize
  - Simultaneous move is conceptually pretty easy, but requires solving an LP at each state
- General sum case inherits some tricky issues from one-shot games
  - Computational difficulty in finding equilibria
  - Equilibrium selection problem
  - No simple generalization of standard MDP algorithms