# Friend-or-Foe Q-learning in General-Sum Games

By Michael L. Littman

Presentation by Paul Pereira

## Definitions

- One-stage general-sum n-player game :
  - -N players
  - -each with their own action set $A_1$, $A_2$, … $A_n$.
  - -and payoff function $R_1$,…, $R_n$

- A policy $P_i$ is the probability distribution over $A_i$

- $R_i$(P1, P2,…,Pn) is the expected return of player i if the n players use policy P1 to Pn.

# Equilibriums

- Nash Equilibrium:
  for each player i:

$$R_i(\pi_1, \ldots, \pi_n) \geq R_i(\pi_1, \ldots, \pi_{i-1}, \pi_i', \pi_{i+1}, \ldots, \pi_n), \tag{1}$$

- Adversarial Equilibrium:
  for each player i:

$$R_i(\pi_1, \ldots, \pi_n) \leq R_i(\pi_1', \ldots, \pi_{i-1}', \pi_i, \pi_{i+1}', \ldots, \pi_n'), \tag{2}$$

- Coordination Equilibrium:
  for each player i:

$$R_i(\pi_1, \ldots, \pi_n) = \max_{a_1 \in A_1, \ldots, a_n \in A_n} R_i(a_1, \ldots, a_n) \tag{3}$$

# Example to illustrate Nash Equilibrium

If row player chooses 0 and column player chooses 1, then we have a coordination equilibrium.

If the row player chooses 1 and the column player chooses 0, then we have an adversarial equilibrium.

$$\text{row:}R_1 = \begin{bmatrix} -1 & 2 \\ 0 & 1 \end{bmatrix}, \text{column:}R_2 = \begin{bmatrix} 1 & 2 \\ 0 & -1 \end{bmatrix}. \quad (4)$$

# Adapting The Single-Layer To MP

- One Q function per player:

$$Q_i(s, a_1, \ldots, a_n) = R_i(s, a_1, \ldots, a_n) + \gamma \sum_{s' \in S} T(s, a_1, \ldots, a_n, s') Q_i(s', \pi_1, \ldots, \pi_n),$$

- Q function can be used to link Markov games and one-stage games. Treat Q functions at each state as payoff functions for individual one stage games

- Policies at individual states are in equilibrium if and only if the overall multistage policies are in equilibrium.

# Nash-Q

- Nash-Q is Q-Learning with an adapted update rule for general sum n player games.

$$Q_i[s, a_1, \ldots, a_n] := (1 - \alpha_t) Q_i[s, a_1, \ldots, a_n] +$$
$$\alpha_t \left( r_i + \gamma \mathsf{Nash}_i(s, Q_1, \ldots, Q_n) \right) \qquad (6)$$

with : $\mathsf{Nash}_i(s, Q_1, \ldots, Q_n) = Q_i(s, \pi_1, \ldots, \pi_n)$

- Requires one Q function per player in order to perform update. Finding Nash Equilibrium can be very expensive operation.

# Conditions for Convergence

- In most general sum games, Nash-Q will not converge to a Nash equilibrium. This is because in a general sum game, the Nash Equilibrium of every state is not unique.
- If a one-stage game has coordination equilibrium, all of its coordination equilibria have the same value
- The same is true for adversarial equilibria (proof on board)
- We can focus on games where these types of equilibria are present.

# Conditions(2)

- In order for Nash-Q to converge, there must be an adversarial or coordination equilibrium for the entire game and all equilibria encountered during learning must be unique.
- Another way to guarantee convergence, assuming we know that the game has an adversarial or coordination equilibrium is to modify the update rule to learn for this specific type of equilibrium.
- This is the Friend-or-Foe Q-Learning Algorithm

# Friend-or-Foe Q-Learning

- This algorithm requires us to say for each player i, which of the other n-1 players are friends and which are foes.
- Friends are players who are assumed to be working with player i to maximize her score.
- Foes are other players who attempt to minimize the score of player i.
- Update rule for two player case:

$$\mathsf{Nash}_1(s, Q_1, Q_2) = \max_{a_1 \in A_1, a_2 \in A_2} Q_1[s, a_1, a_2] \qquad (7) \qquad \text{(friend)}$$

$$\mathsf{Nash}_1(s, Q_1, Q_2) = \max_{\pi \in \Pi(A_1)} \min_{a_2 \in A_2} \sum_{a_1 \in A_1} \pi(a_1) Q_1[s, a_1, a_2] \qquad (8) \qquad \text{(foe)}$$

# Friend-or-Foe Q-Learning

- In the n-player case, we simply assume that the game is two-player zero sum by grouping the interest of the players.
- Update rule in the general case:

$$\mathsf{Nash}_i(s, Q_1, \ldots, Q_n) =$$

$$\max_{\pi \in \Pi(X_1 \times \cdots \times X_k)} \min_{y_1, \ldots, y_l \in Y_1 \times \cdots \times Y_l} \sum_{x_1, \ldots, x_k \in X_1 \times \cdots \times X_k}$$

$$\pi(x_1) \cdots \pi(x_k) Q_i[s, x_1, \ldots, x_k, y_1, \ldots, y_l].$$

- Friend-or-Foe Q-learning converges. This follows from the proof of convergence for minimax-Q.

- If the game has an adversarial equilibrium, Foe-Q learns the values for a Nash Equilibrium policy.

- This follows from the proof that an adversarial equilibrium can be found using the minimax calculation from the update rule

  Proof: We can show that the equilibrium that satisfies the minimax rule has the same return value as an equilibrium that satisfies the adversarial condition

- If the game has a coordination equilibrium, Friend-Q learns the values for this Nash Equilibrium policy (by definition).

# Example

Good for Friend-Q

Good for Foe-Q

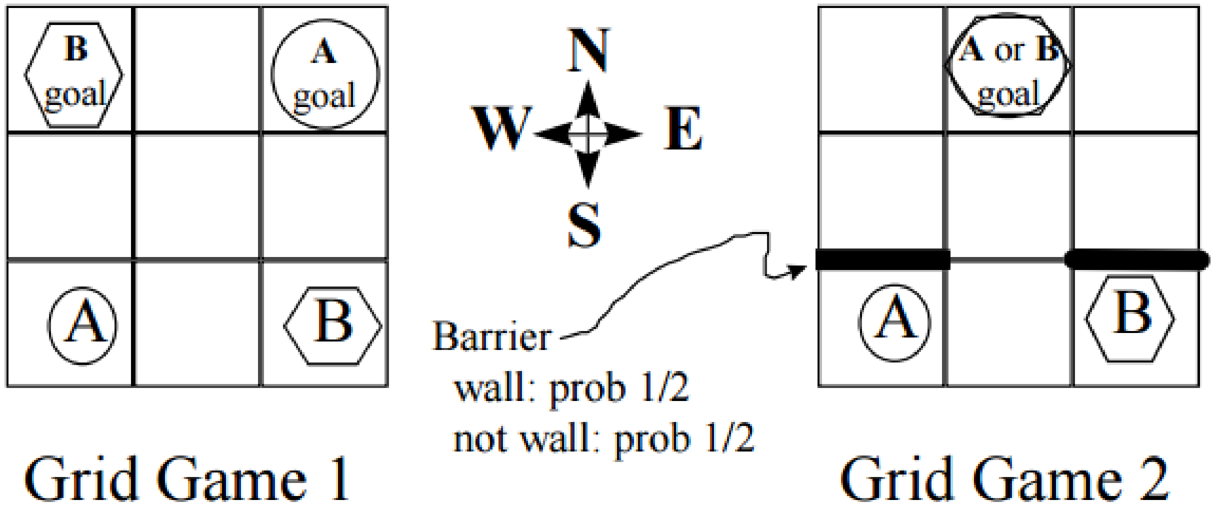

*Figure 1.* Two general-sum grid games.

## Reference

- Littman, M. (2003). Friend-or-Foe Q-learning in General-Sum Games