On-ramp Merging with Multi-Agent Reinforcement Learning

CIS 579: Artificial Intelligence, Fall 2017

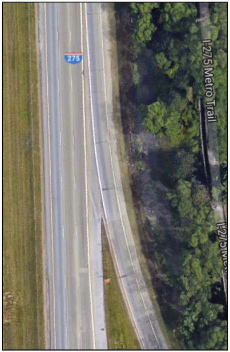
Heng Liu

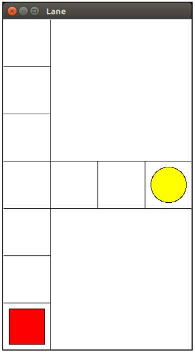
ID: 62774047

Instructor: Dr. Luis E. Ortiz

December 14, 2017

Overview





- Vehicle travels on the on-ramp with the goal of merging with approaching in-lane traffic
- Simplified as two-agent gridworld

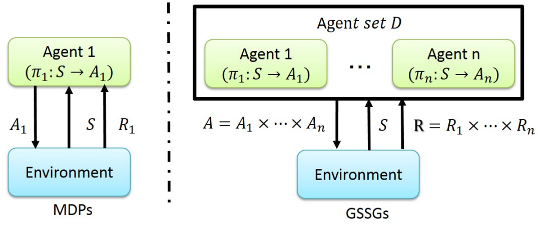
Q-learning[2]:

- Most popular and widely used form of reinforcement learning which determines optimal actions in a Markovian domain
- An iterative approach which learns to improve by repetitive evaluation at particular states

$$Q(s_t, a_t) \leftarrow (1-\alpha) \cdot \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\left(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} \right)}^{\text{learned value}} \tag{1}$$

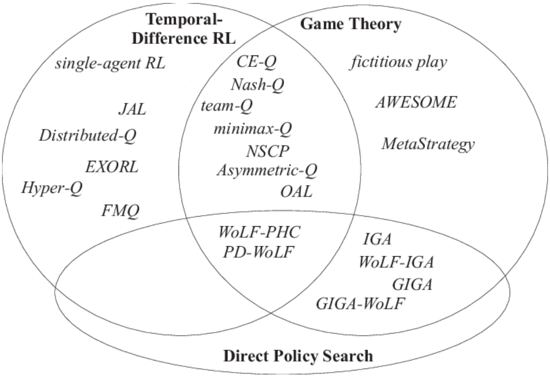
Multi-Agent Reinforcement Learning (MARL)

- Non-stationary environments - other agents acting
- Not an arbitrary stochastic process - other agents be presumed rational
- Game theory adapted to solve multi-agent situations which involve compromises and cooperation
- Stochastic games can be thought as an extension of Markov decision processes in the sense that they deal with multiple agents in a multiple state situation.

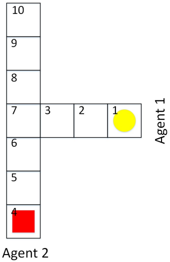


Multi-Agent Reinforcement Learning (MARL)

- Fictitious Play - belief-based learning rule, i.e., players form beliefs about opponent play from the entire history of past play and behave rationally with respect to these beliefs
- Nash-Q[5] tries to address the general problem of learning in two-player general-sum games,



Problem Representation



- Number of agents : n $= 2$
- Action space for agent i:
  $A_i = \{maintain, decelerate, accelerate\}$
- State space: $S = (1,4),(2,5),\ldots$ where a state $s = (l^1, l^2)$ represents the agents joint location.
- Reward function for agent i:

$$f(x) = \begin{cases} 1000 & (L(l^1,a^1) = 10 \textit{ or } L(l^2,a^2) = 10) \textit{ and } L(l^1,a^1) \neq L(l^2,a^2), \textit{successful merge} \\ -10000 & L(l^1,a^1) = L(l^2,a^2) \textit{ or } (l^1 < l^2 \textit{ and } L(l^1,a^1) > L(l^2,a^2)) \textit{ or } (l^1 > l^2 \textit{ and } L(l^1,a^1) < L(l^2,a^2)), \textit{collision} \\ -1 & L(l^i,a^i) - l_i = 1, \textit{decelerate} \\ 0 & L(l^i,a^i) - l_i = 2, \textit{maintain speed} \\ -1 & L(l^i,a^i) - l_i = 3, \textit{accelerate} \end{cases} \tag{2}$$

where $L(l,a)$ is the potential new location resulting from choosing action a in position l.

Fictitious Play

- Each player assumes that his opponent is using a stationary mixed strategy, and updates his beliefs about this stationary mixed strategies at each step
- Players choose actions in each period to maximize that periods expected payoff given their prediction of the distribution of opponents actions, which they form according to:

$$\mu_i^t(s_{-i}) = \frac{\eta_i^t(s_{-i})}{\sum_{s_{-i} \in S_{-i}} \eta_i^t(s_{-i})}$$

i.e., player i forecasts player i's strategy at time t to be the empirical frequency distribution of past play

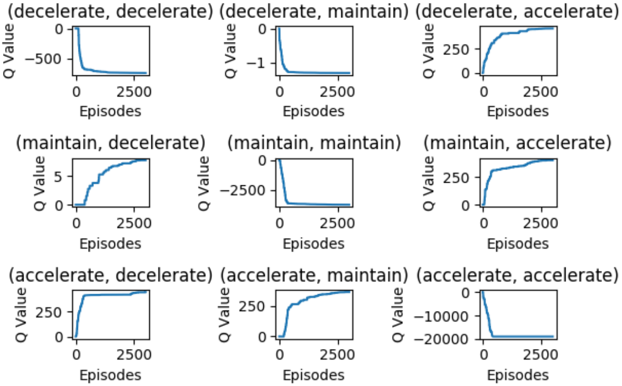


Figure: Fictitious Play Q Values at Starting Position (1, 4)

Results from Fictitious Play

Table: Q-values at state (1, 4) after 3000 episodes

	decelerate	maintain	accelerate
decelerate	-741.16, -741.16	-1.31, 0.00	461.52, 461.52
maintain	7.76, 6.21	-3733.57, -3733.57	401.61, 400.21
accelerate	444.07, 444.07	361.72, 363.08	-19001.90, -19001.90

Table: Q-values at state (2, 7) after 3000 episodes

	decelerate	maintain	accelerate
decelerate	-211.07, -211.07	-0.10, 0.00	521.18, 521.18
maintain	-186.39, -186.75	-6190.93, -6190.93	409.51, 409.10
accelerate	-6216.83 , -6216.83	-851.73, -851.24	521.18, 521.18

Results from Fictitious Play

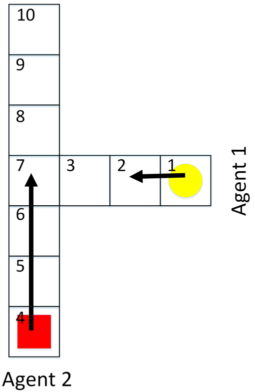


Figure: Fictitious Play paths at (1, 4)

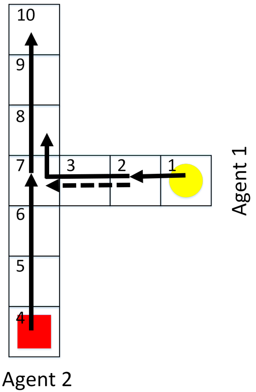


Figure: Fictitious Play paths at (2, 7)

Nash Q-Learning

Initialize:

Let $t = 0$, get the initial state s_0.

Let the learning agent be indexed by i.

For all $s \in S$ and $a^j \in A^j$, $j = 1, \ldots, n$, let $Q_t^j(s, a^1, \ldots, a^n) = 0$.

Loop

Choose action a_t^i.

Observe $r_t^1, \ldots, r_t^n; a_t^1, \ldots, a_t^n$, and $s_{t+1} = s'$

Update Q_t^j for $j = 1, \ldots, n$

$$Q_{t+1}^j(s, a^1, \ldots, a^n) = (1 - \alpha_t) Q_t^j(s, a^1, \ldots, a^n) + \alpha_t [r_t^j + \beta NashQ_t^j(s')]$$

where $\alpha_t \in (0, 1)$ is the learning rate, and $NashQ_t^k(s')$ is defined in (7)

Let $t := t + 1$.

where $NashQ_t^i(s')$ is agent is payoff in state s' for the selected equilibrium. Nash equilibria is solved using 'support enumeration' algorithm[8].

Results from Nash Q-Learning

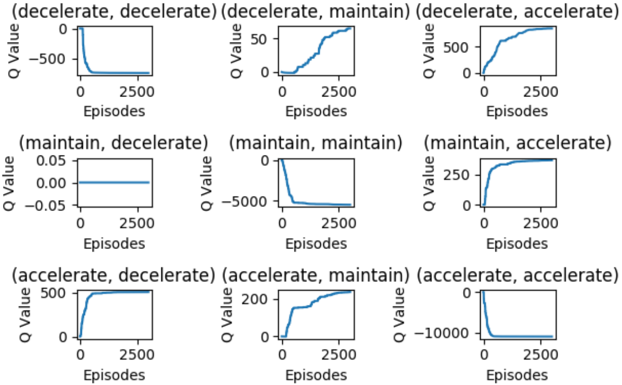


Figure: Nash Q Values at Starting Position (1, 4)

Results from Nash Q-Learning

Table: Q-values in state (1, 4) after 3000 episodes

	decelerate	maintain	accelerate
decelerate	-743.09, -743.09	65.22, 67.13	851.05, 851.05
maintain	0.00, -1.31	-5524.48, -5524.48	366.65, 365.29
accelerate	506.10, 506.10	236.09, 237.33	-10897.53, -10897.53

Table: Q-values in state (2, 7) after 3000 episodes

	decelerate	maintain	accelerate
decelerate	-214.77, -214.77	264.17, 265.38	946.72, 946.72
maintain	-1790.13, -1791.25	-4125.34, -4125.34	911.37, 910.46
accelerate	-8452.73, -8452.73	-3478.32, -3477.04	999.00, 999.00

Results from Nash Q-Learning

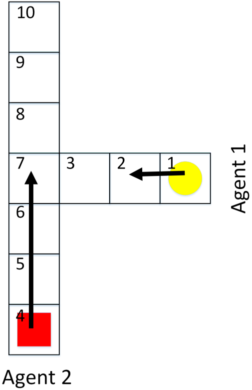


Figure: Nash equilibrium paths at (1, 4)

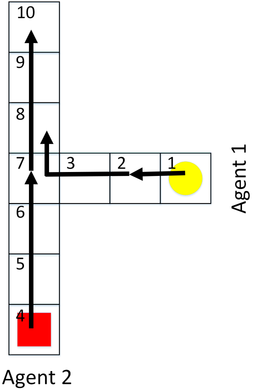


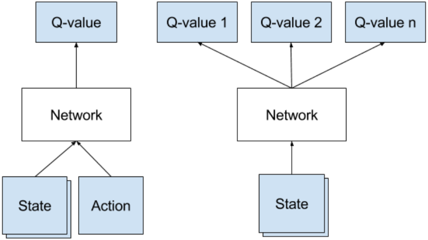
Figure: Nash equilibrium paths at (2, 7)

Simulation

- TkInter. Python's standard GUI package, used for gridworld simulation.
- Demo.

Future Works

- Add more agents to increase complexity
- Other MARL algorithms to be explored
  - Complexity (interaction between all other agents and all the time)
  - Convergence (Nash Q-learning has certain restrictions)
- Deep Q-Learning with Q-function approximation considering vehicles' state space and action space are actually continuous



References I

[1] Busoniu, Lucian, Robert Babuska, and Bart De Schutter. "A comprehensive survey of multiagent reinforcement learning." *IEEE Transactions on Systems, Man, And Cybernetics-Part C: Applications and Reviews, 38 (2), 2008* (2008).

[2] Watkins, Christopher JCH, and Peter Dayan. "Q-learning." *Machine learning* 8.3-4 (1992): 279-292.

[3] Littman, Michael L. "Markov games as a framework for multi-agent reinforcement learning." Proceedings of the eleventh international conference on machine learning. Vol. 157. 1994.

[4] Hu, Junling, and Michael P. Wellman. "Multiagent reinforcement learning: theoretical framework and an algorithm." *ICML*. Vol. 98. 1998.

[5] Hu, Junling, and Michael P. Wellman. "Nash Q-learning for general-sum stochastic games." *Journal of machine learning research 4*.Nov (2003): 1039-1069.

[6] Neto, Gonalo. "From single-agent to multi-agent reinforcement learning: Foundational concepts and methods." *Learning theory course* (2005).

[7] Wang, Pin, and Ching-Yao Chan. "Formulation of Deep Reinforcement Learning Architecture Toward Autonomous Driving for On-Ramp Merge." *arXiv preprint arXiv:1709.02066* (2017).

References II

[8] Porter, Ryan, Eugene Nudelman, and Yoav Shoham. "Simple search methods for finding a Nash equilibrium." *Games and Economic Behavior* 63.2 (2008): 642-662.

[9] Gaskett, Chris, David Wettergreen, and Alexander Zelinsky. "Q-learning in continuous state and action spaces." *Advanced Topics in Artificial Intelligence* (1999): 417-428.

[10] Tan, Ming. "Multi-agent reinforcement learning: Independent vs. cooperative agents." *Proceedings of the tenth international conference on machine learning* (1993): 330-337.

[11] Hu, Yujing, Yang Gao, and Bo An. "Learning in multi-agent systems with sparse interactions by knowledge transfer and game abstraction." *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2015.