STACKELBERG POLICY GRADIENT: EVALUATING THE PERFORMANCE OF LEADERS AND FOLLOWERS

Quoc-Liem H. Vu, Zane Alumbaugh[1], Ryan Ching, Peter Ding, Arnav Mahajan, Benjamin J. Chasnov, Samuel A. Burden, Lillian J. Ratliff
Department of Electrical and Computer Engineering
University of Washington
Seattle, WA, USA
{qliemvu,chingry,quancd,am172,bchasnov,sburden,ratliffl}@uw.edu
[1]zanedma@gmail.com

ABSTRACT

Hierarchical order of play is an important concept for reinforcement learning to understand better the decisions made by strategic agents in a shared environment. In this paper, we compare the learning dynamics between Stackelberg and simultaneous reinforcement learning agents. Agents are trained using their policy gradient and are tested against each other in a tournament. We compare agent performance in zero-sum and non-zero-sum Markov games. We show that the Stackelberg leader performs better in training under the same parameters. However, under the same parameters in the tournament setting, Stackelberg leaders and followers performed similarly to the simultaneous player. Analytically, hierarchical training can potentially provide stronger guarantees for policy gradient.

1 INTRODUCTION

An emerging field of machine learning pits strategic agents against each other in competitive games. A popular example of the success of competitive machine learning is Google's AlphaZero vs. Stockfish in a zero-sum game of chess. (Dai & Song, 2018; Zhang et al., 2021) Although Stockfish calculates roughly 1000 times more moves per second than AlphaZero, each agent is evenly matched. AlphaZero uses reinforcement learning (RL) to determine the best moves to make given the state of the environment. Motivated by the interest in RL and hierarchical learning, we aim to compare the performance of agents trained via simultaneous and Stackelberg reinforcement learning in both competitive (zero-sum) and general (non-zero-sum) environments.

An important application of RL and game theory involves a hierarchical order of play, which are settings best modeled as Stackelberg games. The concept of the Stackelberg equilibrium (Von Stackelberg, 2010) generalizes the min-max solution to general-sum games. In the two-player scenario, one player acts as the leader who selects an action given the knowledge that the other player (follower) plays a best-response. This viewpoint has long been researched from a control perspective on games (Başar & Olsder, 1998) and in the bilevel optimization community (Danskin, 1966).

The current machine learning perspective on games with a hierarchical decision-making structure mainly focuses on zero-sum games. Theoretical work relates that all stable critical points of simultaneous gradient descent with a timescale separation between players approaching infinity satisfy sufficient conditions for a local Stackelberg equilibrium (Jin et al., 2019).

1.1 RELATED WORKS

The theoretical results of multi-agent reinforcement learning are reviewed in (Zhang et al., 2021). The framework of Markov games (Shapley, 1953; Littman, 1994) has been used to develop learning algorithms. Policy gradient with function approximation (Sutton et al., 1999) is used to implement policies that map states to actions. It is known that the Nash equilibrium always exists for finite-space infinite-horizon discounted competitive Markov games (Filar & Vrieze, 2012), but may not be unique in general. Algorithms have been developed for general sum games (Hu & Wellman, 2003). Reinforcement learning in games with asymmetry like hierarchy amongst agents are explored in (Könönen, 2004; Pereira, 2020).

Our work exists in the context of other research on learning dynamics for multi-agent systems where agents model the learning process of opponents. These models are used to derive various learning updates using gradients (Foerster et al., 2017). Games involving a hierarchical order of play are ubiquitous in nature but are a less explored area of RL. Where previous works focused on implicit learning dynamics for improving the performance of learning algorithms (Fiez et al., 2020; Zheng et al., 2021), we focus on comparing the performance of reinforcement learning agents that adopt a leader or follower role against players trained without hierarchy. We conducted several experiments with agents trained via simultaneous and Stackelberg policy gradient agents by varying learning rates, batch sizes, and neural network hidden layers in zero-sum and non-zero-sum Markov games to understand hierarchy of play dynamics in game type.

Contributions We show that hierarchical training has a benefit to agent performance. In a non-zero-sum Markov game, the Stackelberg leader and follower generally performed better than the simultaneous player. In the zero-sum Markov game, the same trend occurred, with the addition that there was a guarantee that the leader performed better than the follower. In comparison, the first or second player could interchange in performance with the simultaneous player. There is also potential to guarantees for policy gradient. Our work is a preliminary investigation into evaluating the performance of the Stackelberg policy gradient in two-player games. Future work is proposed in the discussion section.

2 PRELIMINARIES

We provide standard mathematical preliminaries for multi-agent reinforcement learning. We consider a two-player fully observable Markov game defined by tuple $\mathcal{G} = (\mathcal{S}, \mathcal{A}^1, \mathcal{A}^2, P, r^1, r^2)$ where $\mathcal{S}$ is the state space, $s \in \mathcal{S}$ is a state, player $i \in \{1, 2\}$, A^i is the player i's action space with $a^i \in \mathcal{A}^i$. The transition kernel is $P : \mathcal{S} \times \mathcal{A}^1 \times \mathcal{A}^2 \to \mathcal{S}$ such that $P(s'|s, a^1, a^2)$ is the probability of transitioning to state s' given that the previous state was s and the agents took actions (a^1, a^2) simultaneously in state s. The reward $r^i : \mathcal{S} \times \mathcal{A}^1 \times \mathcal{B}^2 \to \mathbb{R}$ is the reward function for player i. Note that for zero-sum games, $r^1 \equiv -r^2$, and for non-zero-sum games, the rewards may be anti-aligned. Each agent uses a stochastic policy $\pi^i_{\theta_i}$ parameterized by θ_i.

The cumulative rewards of agent i given a trajectory $\tau = (s_0, a^1_0, a^2_0, \ldots, a^2_T)$ is defined as $R(\tau) = \sum_{t=0}^{T} \gamma^t r(s_t, a^1_t, a^2_t)$ with discount factor $0 < \gamma \leq 1$. The expected cumulative rewards of policy $\pi = \{\pi^1_{\theta_1}, \pi^2_{\theta_2}\}$ is $Q_\theta(s_t, a^1_t, a^2_t) = \mathbb{E}[\sum_{t'=t}^{T} \gamma^{t'-t} r(s_{t'}, a^1_{t'}, a^2_{t'}|s_t, a^1_t, a^2_t)]$. The objective for each player is the expected return given by $J_i(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}[\sum_{t=0}^{T} \gamma^t r(s_t, a^1_t, a^2_t)]$. Agents aim to maximize their expected return.

3 TRAINING AGENTS USING POLICY GRADIENT

We train agents using policy gradient algorithms based on simultaneous and Stackelberg updates. In this section, we provide descriptions of the two policy gradient updates. Algorithm 1 summarizes the training procedures.

3.1 SIMULTANEOUS POLICY GRADIENT

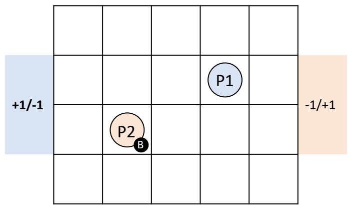
For the simultaneous policy gradient algorithm (`simgrad`), players compute their gradient without extra information of the other players. Each player optimizes its objective

$$\max_{\theta_i \in \Theta_i} J_i(\theta_i, \theta_{-i}),\ i \in \{1, 2\}$$

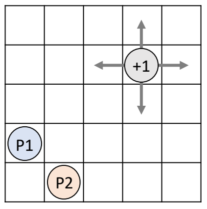
by descending individual policy gradients

$$\nabla_{\theta_i} J_i(\theta) = \mathbb{E}[\nabla_{\theta_i} \log \pi^i_{\theta_i}(s, a^i) \cdot Q_\theta(s, a)]. \tag{1}$$

We direct the readers to the review paper (Zhang et al., 2021) for details about running policy gradient for two players in the mixed (non-zero-sum) setting.



(a) Markov Soccer (zero-sum)



(b) Grid Hunter (non-zero-sum)

Figure 1: **Experiment setup.** In the grid world environments, agents observe the full state of the environment and choose individual actions. ***A:*** In Markov Soccer, players start on opposite ends of the center, with a randomized initial position and possession of the ball. A reward is given when an agent reaches the opponent's goal and minor rewards for ball possession. The available actions to the agents is to move in the four cardinal directions. If an agent without possession of the ball chooses to move where the agent with possession of the ball, the ball possession transfers. The agent that stole the ball is now where the previous agent is and the other agent moves to whichever location they decided. ***B:*** In Grid Hunter, two pursuers start in a corner of the grid with random placement of the prey. The prey moves to avoid the two hunters while the hunters move to catch the prey. The prey and hunters is able to move in the four cardinal directions. The prey movement is random while the hunter movement is learned. A reward is given to the agent who first reaches the prey.

Algorithm 1 Simultaneous or Stackelberg policy gradient (`simgrad` or `stackgrad`)

```
Require: Markov game $\mathcal{G}$, parameters $T, K, \gamma_1, \gamma_2 > 0$ and $(\theta_1^0, \theta_2^0)$.
1: for $t = 0;\ t < T$ do
2:     Sample a batch of $K$ trajectories from $\mathcal{G}$ using policies $\pi^1_{\theta_1^t},\ \pi^2_{\theta_2^t}$
3:     if simgrad then
4:         $\theta_1^{t+1} = \theta_1^t + \gamma_1 \nabla_{\theta_1} J_1(\theta_1^t, \theta_2^t)$        Equation (1)
5:     else if stackgrad then
6:         $\theta_1^{t+1} = \theta_1^t + \gamma_1 \nabla_{\theta_1} J_1(\theta_1^t, r^*(\theta_1^t))$        Equation (2)
7:     end if
8:     $\theta_2^{t+1} = \theta_2^t + \gamma_2 \nabla_{\theta_2} J_2(\theta_1^t, \theta_2^t)$        Equation (1)
9: end for
```

3.2 Stackelberg policy gradient

For the Stackelberg policy gradient algorithm (`stackgrad`), the leader of the game knows the other player's response and incorporates this response in its gradient update. In a Stackelberg game, the leader and follower aim to solve the following optimization problems

$$\max_{\theta_1 \in \Theta_1} \{J_1(\theta_1, \theta_2) | \theta_2 \in \arg\max_{y \in \Theta_2} J_2(\theta_1, y)\}, \quad \max_{x_2 \in X_2} J_2(x_1, x_2)$$

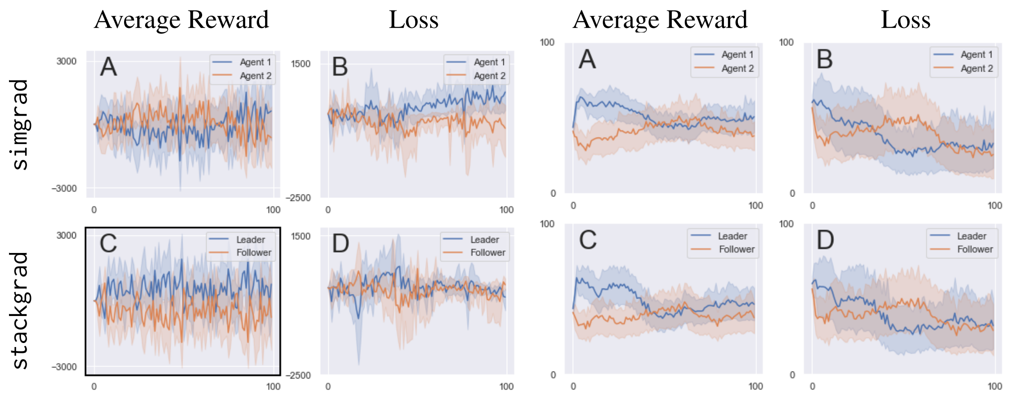
where the leader uses its total derivative

$$\nabla_{\theta_1} J_1(\theta_1, r^*(\theta_1)) = \left(\nabla_{\theta_1} J_1(\theta) - \nabla_{\theta_1} \nabla_{\theta_2} J_2(\theta) \left[\nabla^2_{\theta_2} J_2(\theta)\right]^{-1} \nabla_{\theta_2} J_1(\theta)\right) \tag{2}$$

and the follower has response $r^* : \Theta_1 \to \Theta_2$ and uses its individual policy gradient. We direct the readers to the papers (Fiez et al., 2020; Yang et al., 2022) for more details on how to compute the leader's update using the chain rule and compute the inverse operation by conjugate gradient.

3.3 Training results

We perform a grid search over learning rates $\gamma_1, \gamma_2 \in \{10^{-1}, 10^{-2}, 10^{-3}\}$, batch sizes $K \in \{1000, 5000\}$, and feed-forward neural network hidden layers using one-layer with $32,\ 64,\ 128$ neurons per layer and one three-layer parameter, $[128, 64, 32]$. These parameters total 24 different agent configurations per policy gradient. We run both policy gradient methods for a total of 100 epochs.



(a) Markov Soccer (zero-sum) (b) Grid Hunter (non-zero-sum)

Figure 2: **Training performance.** The leader trained using the Stackelberg policy gradient receives a higher average reward than the follower in the zero-sum game (bold outline). We ran with one seed for each of the 24 different parameter configurations. We plot the agents' average rewards and losses during training. The shaded regions show the 95% standard deviation bound of results. The x-axis represents the number of epochs. **A:** `simgrad` average rewards (larger is better) **B:** `simgrad` average loss (smaller is better) **C:** `stackgrad` average rewards **D:** `stackgrad` average loss.

The Stackelberg leader performs better in Markov Soccer than the follower during training, as shown in Figure 2(a)C. The average magnitude of return for the Stackelberg leader was 488. In contrast, the performance of `simgrad` seems to oscillate randomly as shown in Figure 2(a)A, with a smaller average magnitude of return of 96.4. This shows that the Stackelberg leader is capable of anticipating the follower's response during training to achieve a higher reward.

We see a similar reward and loss behavior in Grid Hunter for both algorithms. As shown in Figure 2(b). For `simgrad` player 1, the average return was 51.1, and for `simgrad` player 2, the average return was 41.3. Similarly, for Stackelberg player 1, the average return was 48.5, and for `simgrad` player 2, the average return was 38.6. This trend continues with the losses as shown in Figure 2bb and Figure 2bd. For `simgrad` player 1, the average loss was 39.7, and for `simgrad` player 2, the average loss was 39.1. For Stackelberg player 1, we saw an average loss of 37.4, and for Stackelberg player 2, 38.6. Because Grid Hunter is a general sum game, there may be no strong guarantees of convergence. Furthermore, the learning dynamics exhibit some behaviors of cycling.

4 Testing Agents using Tournaments

We test the agents' performance against each other by fixing the parameters of the agent's policies and running a tournament-style competition. The competition allows us to compare agents with different parameters and neural network models. It also allows us to compare leaders and followers.

The motivation for running tournaments is to determine how well the agents perform relative to each other. The resulting "test matrix" can be further analyzed to determine the agent rankings. In this preliminary work, we report the average results from the tournament. Future work will involve how to better understand the output of the test matrix.

4.1 Playing against a leader or follower

To test the leader's performance, we pit player 1 trained with `stackgrad` against player 2 trained with `simgrad`. Futhurmore, to test the follower's performance, we pit player 2 trained with `stackgrad` against player 1 trained with `simgrad`. Finally, to test all agents' performance, we pit all agents against an agent performing random actions. The agents' relative performance in the game (i.e. which agent achieves a higher reward) is used to evaluate the agent's performance relative to other agents trained with different parameters or algorithms.

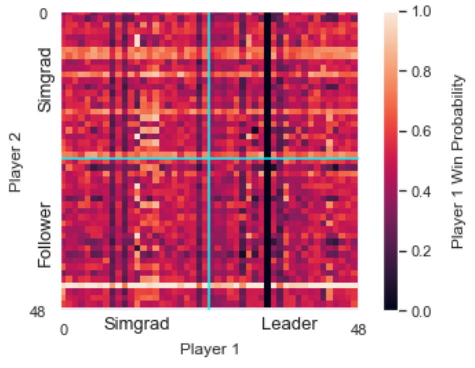
P2 \ P1	simgrad	leader	random
simgrad	52.3%	48.9%	31.6%
follower	48.8%	45.0%	31.9%
random	66.4%	66.5%	50.0%

(a) Markov Soccer

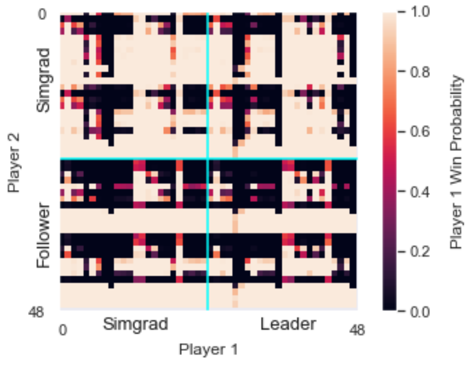
P2 \ P1	simgrad	leader	random
simgrad	70.1%	53.3%	40.5%
follower	67.5%	52.8%	40.4%
random	74.5%	74.5%	50.0%

(b) Grid Hunter

Table 1: **Tournament results.** Percentage of games where player 1 achieves a higher reward than player 2. ***A:*** In Markov Soccer, we see similar performance between `simgrad` and Stackelberg agents. With similar results against random agents, and similar results when competing against each other. The `simgrad` player 1 performed better than its player 2 counterpart. In contrast, the Stackelberg follower performed better than the Stackelberg leader counterpart. ***B:*** In Grid Hunter, we see dominant performance of the `simgrad` player 1 against `simgrad` player 2 and the Stackelberg follower. The Stackelberg leader had slightly better performance than the `simgrad` player 2 and its Stackelberg follower counterpart. Similar to Markov Soccer, both algorithms had similar performance against the random agent.



(a) Markov Soccer



(b) Grid Hunter

Figure 3: **Test performance.** The test matrix of agents after playing a tournament in each environment. The horizontal and vertical axes are indices for players 1 and 2, respectively. The color represents the probability of player 1 receiving a higher score than player 2 when sampled from 500 episodes. ***A:*** In Markov Soccer, performance was similar between all agents. Most players are equally likely to win, although some choices of parameters performed worse or better. ***B:*** In Grid Hunter, agents with certain parameters performed better than all other agents.

4.2 Playing against random agents

We employed agents performing random moves to test agent performance against a baseline. Generally, the `simgrad` and Stackelberg agents performed similarly better than the random agents. This comparison against a baseline also revealed agent parameters that were not as strong or hyper performers. As well as this, to test the validity of our environments and sample size, we competed two random agents. As shown in Table 1, the results in both environments, the random agents player 1 had a 50.0% win percentage, proving symmetry of the environment.

4.3 Test results

Competitive tournaments revealed the relative performance between agents. As shown in Table 1, `simgrad` and Stackelberg performance as either player 1 or 2 against random agents was almost identical. While against each other, player 2 of `simgrad` and Stackelberg performed similarly better than the player 1. Interestingly, where `simgrad` player 1 performed better than the player 2, Stackelberg was the opposite with a better performing follower than the leader.

In Markov Soccer, performance was similar between all agents. However, one Stackelberg agent did notably worst than the rest (batch size = 5000, learning rate = 0.1, neural net of size [128,64,32]) as shown as the distinct streak in Figure 3a. The best performing agents were `simgrad` with batch sizes = 1000 or 5000, learning rate = 0.001 and neural net of size 128. Similar parameters for the Stackelberg agents were close behind.

In Grid Hunter, agents with either policy gradient with a high batch size of 5000, learning rate of 0.01, and a neural net of size [64] highly performed against all other agents, likely due to the high-dimensionality of the environment. Close behind were agents with batch size of 5000, learning rate of 0.001, and neural net of size [128], as shown as the distinctly bright regions of Figure 3b. The worst performing agents similarly had a learning rate of 0.1.

5 Discussion

In this paper, we study the simultaneous and Stackelberg policy gradient algorithms and the performance of agents trained with them. Our preliminary experiments show that leaders achieve a larger average reward than followers for zero-sum games during training. On the other hand, in general sum games, the leader did not show a significant advantage over the follower. More future work is needed to examine the leader and followers' learned strategies in general-sum games. For example, we can perform analyses not only of their average rewards, but also of their trajectories through the environment. In addition, we can compare these learned strategies to that of a fully-cooperative general-sum game with team based reward setting.

A further investigation into how the Stackelberg learning dynamics affect the quality of the solutions is needed. Other experiments can be devised to determine whether hierarchical training is beneficial for particular types of environments and agent reward structures. Although our results found good parameters for each environment, we can extend our work to find more suitable hyper parameters. In particular, the learning rate ratio γ_2/γ_1 is a important parameter to tune, as it represents the timescale separation between the agents. We want the follower to be attracted to its equilibrium at a faster timescale than the leader.

In future work, we will test these learning updates on a wider range of environments. Currently, our simple experiments serve as a testing ground for the algorithm. We plan to expand the domain of environments and test our algorithm on other developed multi-agent gym packages.

Finally, we can further quantify the difference between agent performance under different hyper parameters using a method for calcuating ELO scores (Gosiewska, 2019). Other works in RL has utilized ELO scores in comparing agent models (Wang, 2021) as well as the addition of a Bayesian algorithm to understand opponent behavior more (Yang, 2019). These extensions could provide a new lens to evaluate the performance of agents trained using the Stackelberg policy gradient algorithm.

6 Conclusion

Understanding the dynamics of Stackelberg games is an essential step for translating RL to more comprehensive applications. An important aspect of this understanding is the performance between simultaneous and Stackelberg players. In training the agents, we saw that the Stackelberg leader rewards tend to be higher than the Stackelberg follower. Furthermore, in competitive tournaments in zero-sum games, we saw similar performance between `simgrad` and Stackelberg agents. In contrast to non-zero-sum games, we saw stronger performing player 1's than player 2's. In conclusion, we show that agents that undergo hierarchical training can be beneficial to its performance when playing against other agents.

Acknowledgments

This research was supported in part by National Science Foundation Awards #1836819, #2045014, #1844729

References

Tamer Başar and Geert Jan Olsder. *Dynamic Noncooperative Game Theory*. SIAM, 1998.

Shaw A. Li L. Xiao L. He N. Liu Z. Chen J. Dai, B. and L. Song. Sbeed: Convergent reinforcement learning with nonlinear function approximation. *International Conference on Machine Learning*, 38(139):1125–1134, 2018.

John M Danskin. The theory of max-min, with applications. *SIAM Journal on Applied Mathematics*, 14(4):641–664, 1966.

Tanner Fiez, Benjamin Chasnov, and Lillian Ratliff. Implicit learning dynamics in stackelberg games: Equilibria characterization, convergence analysis, and empirical study. In *International Conference on Machine Learning*, pp. 3133–3144. PMLR, 2020.

Jerzy Filar and Koos Vrieze. *Competitive Markov Decision Processes*. Springer Science & Business Media, 2012.

Jakob N. Foerster, Richard Y. Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. Learning with opponent-learning awareness. *CoRR*, abs/1709.04326, 2017. URL `http://arxiv.org/abs/1709.04326`.

Woznica Zwolinkski Biecek Gosiewska, Bakala. Epp: interpretable score of model predictive power. *Journal of Machine Learning Research*, 4(Nov), 2019.

Junling Hu and Michael P Wellman. Nash q-learning for general-sum stochastic games. *Journal of Machine Learning Research*, 4(Nov):1039–1069, 2003.

Ville Könönen. Asymmetric multiagent reinforcement learning. *Web Intelligence and Agent Systems: An international journal*, 2(2):105–121, 2004.

Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine Learning Proceedings 1994*, pp. 157–163. Elsevier, 1994.

Sahil Pereira. Stackelberg multi-agent reinforcement learning for hierarchical environments. Master's thesis, University of Waterloo, 2020.

Lloyd S Shapley. Stochastic games. *Proceedings of the National Academy of Sciences*, 39(10): 1095–1100, 1953.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in Neural Information Processing Systems*, 12, 1999.

Heinrich Von Stackelberg. *Market Structure and Equilibrium*. Springer Science & Business Media, 2010.

Qi Peng Tang Zhang Li Pi He Gao Long Yuan Wang, Song. Scc: an efficient deep reinforcement learning agent mastering the game of starcraft ii. *International Conference on Machine Learning*, 38(139), 2021.

Boling Yang, Liyuan Zheng, Lillian J Ratliff, Byron Boots, and Joshua R Smith. Stackelberg maddpg: Learning emergent behaviors via information asymmetry in competitive games. 2022.

Meng Zhang Zheng Zheng Yang, Hao. Towards efficient detection and optimal response against sophisticated opponents. *International Joint Conference on Artificial Intelligence*, 28(Nov):623–629, 2019.

Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of Reinforcement Learning and Control*, pp. 321–384, 2021.

Liyuan Zheng, Tanner Fiez, Zane Alumbaugh, Benjamin Chasnov, and Lillian J Ratliff. Stackelberg actor-critic: A game-theoretic perspective. In *AAAI Workshop on Reinforcement Learning in Games*, 2021.