Reinforcement Nash Equilibrium Solver

Extended Abstract

Xinrun Wang*
Nanyang Technological University
Singapore
xinrun.wang@ntu.edu.sg

Chang Yang*
The Hong Kong Polytechnic University, Hong Kong SAR, China
22124101r@connect.polyu.hk

Shuxin Li
Nanyang Technological University
Singapore
shuxin.li@ntu.edu.sg

Pengdeng Li
Nanyang Technological University
Singapore
pengdeng.li@ntu.edu.sg

Xiao Huang
The Hong Kong Polytechnic University, Hong Kong SAR, China
xiaohuang@comp.polyu.edu.hk

Hau Chan
University of Nebraska-Lincoln
Lincoln, Nebraska, United States
hchan3@unl.edu

Bo An
Nanyang Technological University
Singapore
boan@ntu.edu.sg

ABSTRACT

Nash Equilibrium (NE) is the canonical solution concept of game theory, which provides an elegant tool to understand the rationalities. Computing NE in two- or multi-player general-sum games is PPAD-Complete. Therefore, in this work, we propose REinforcement Nash Equilibrium Solver (RENES), which *trains a single policy to modify the games with different sizes and applies the solvers on the modified games where the obtained solution is evaluated on the original games*. Specifically, our contributions are threefold. i) We represent the games as α-rank response graphs and leverage graph neural network (GNN) to handle the games with different sizes as inputs; ii) We use tensor decomposition, e.g., canonical polyadic (CP), to make the dimension of modifying actions fixed for games with different sizes; iii) We train the modifying strategy for games with the widely-used proximal policy optimization (PPO) and apply the solvers to solve the modified games, where the obtained solution is evaluated on original games. Extensive experiments on large-scale normal-form games show that our method can further improve the approximation of NE of different solvers, i.e., α-rank, CE, FP and PRD, and can be generalized to unseen games.

KEYWORDS

Game Theory, Reinforcement Learning, Generalizability

ACM Reference Format:
Xinrun Wang*, Chang Yang*, Shuxin Li, Pengdeng Li, Xiao Huang, Hau Chan, and Bo An. 2024. Reinforcement Nash Equilibrium Solver: Extended Abstract. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 3 pages.

*Equal contribution.

Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand. © 2024 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

1 INTRODUCTION

Game theory provides a pervasive framework to model the interactions between multiple players [6]. The canonical solution concept in non-cooperative games, i.e., the players try to maximize their own utility, is Nash Equilibrium (NE), where no player can change its strategy unilaterally to increase its own utility [13]. According to Roger Myerson, the introduction of NE is a watershed event for game theory and economics [12]. NE provides an impetus to understand the rationalities in much more general economic contexts and lies at the foundation of modern economic thoughts [7, 12]. Mixed strategy NE exists in any game with finite players and actions [13]. However, from an algorithmic perspective, computing NE in two-player or multi-player general-sum games is PPAD-Complete [4, 5]. In two-player zero-sum games, NE can be computed in polynomial time via linear programming. In more generalized cases, the Lemke–Howson algorithm is the most recognized combinatorial method [10], while using this algorithm to identify any of its potential solutions is PSPACE-complete [7].

To address the above issues, we propose REinforcement Nash Equilibrium Solver (RENES). Our main contributions are three-fold. First, we represent the games with different sizes as α-rank response graphs, which are used to characterize the intrinsic properties of games [14], and then leverage the graph neural network (GNN) to take the α-rank response graphs as inputs. Second, we use tensor decomposition, e.g., canonical polyadic (CP), to make the modifying actions fixed for games with different sizes, rather than changing a payoff value once. Third, we train the modifying strategy for games with the widely-used proximal policy optimization (PPO) and apply the solvers to solve the modified games, where the obtained solution is evaluated on original games. Extensive experiments on large-scale normal-form games, i.e., 3000 sampled games for training and 500 sampled games for testing, show that our method can further improve the approximation of NE of different solvers, i.e., α-rank, CE, FP and PRD, and can be generalized to unseen games. To the best of our knowledge, this work is the first effort in game theory that leverages RL methods to train a single strategy for modifying the games to improve the solvers' approximation performances.

2 PRELIMINARIES

Consider the K-player normal-form game, where each player $k \in [K]$ has a finite set of actions $\mathcal{A}^k$. We use $\mathcal{A}^{-k}$ to represent the action space excluding the player k, also for other terms. We denote the joint action space as $\mathcal{A} = \times_{k\in[K]}\mathcal{A}^k$. Let $\boldsymbol{a} \in \mathcal{A}$ be the joint action of K players and $M(\boldsymbol{a}) = \langle M^k(\boldsymbol{a})\rangle \in \mathbb{R}^K$ is the payoff vector of players when playing the action $\boldsymbol{a}$. A mixed strategy profile is defined as $\pi \in \Delta(\mathcal{A})$, which is a distribution over $\mathcal{A}$ and $\pi(\boldsymbol{a})$ is the probability that the joint action $\boldsymbol{a}$ will be played. The expected payoff of player $k \in [K]$ is denoted as $M^k(\pi) = \sum_{\boldsymbol{a}\in\mathcal{A}} \pi(\boldsymbol{a})M^k(\boldsymbol{a})$. Given a mixed strategy π, the best response of player $k \in [K]$ is defined as $\text{BR}^k(\pi) = \arg\max_{\mu\in\Delta(\mathcal{A}^k)}[M^k(\mu, \pi^{-k})]$. A factorized mixed strategy $\pi(\boldsymbol{a}) = \prod_{k\in[K]} \pi^k(a^k)$ is Nash Equilibrium (NE) if $\pi^k \in \text{BR}^k(\pi)$ for $k \in [K]$. We define the NashConv value as $\texttt{NC}(\pi) = \sum_{k\in[K]} M^k(\text{BR}^k(\pi), \pi^{-k}) - M^k(\pi)$ to measure the distance of the mixed strategy from an NE. Computing NE in general-sum games is PPAD-Complete [5].

3 RENES

We introduce the proposed REinforcement Nash Equilibrium Solver (RENES). The general procedure of RENES is displayed in Figure 1.

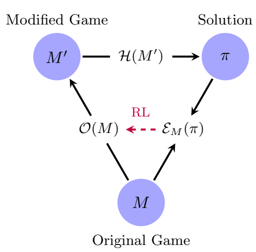


Figure 1: Flow of RENES.

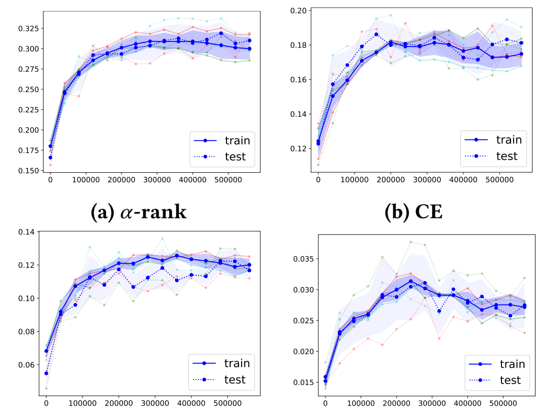
To handle the games with different sizes, we represent the games as the α-rank response graphs, which is shown to represent the intrinsic properties of games in [14], and then use graph neural network (GNN) [8, 16, 17] to extract the features of games. We note that GNN can efficiently handle the graphs with different sizes [11], as it takes the neighboring information to update the node embeddings. For the action space, we consider a more compact action space with tensor decomposition [9]. Specifically, we use the canonical polyadic (CP) decomposition of the payoff table M and set the rank r to be fixed and the action of RENES is the coefficients over r:

$$M \approx \sum_{i=1}^{r} \lambda_i \cdot m_{1,i} \otimes m_{2,i} \otimes \cdots \otimes m_{K+1,i}, \tag{1}$$

where $\boldsymbol{\lambda} = \langle\lambda_i\rangle, i = 1, \cdots, r$ are the weights of the decomposed tensors and $m_{k,i}, k \in \{1, \ldots, K+1\}$ are the factors which are used to modify the game. For the decomposition, the weight $\boldsymbol{\lambda} = \mathbf{1}$[1]. Given any arbitrary weight $\boldsymbol{\lambda}$, we can reconstruct the payoff tensor with the reconstruction oracle $\mathcal{R}_M(\boldsymbol{\lambda})$. Therefore, we let the modified oracle $\mathcal{O}$ to modify the weights and update the game by

$$M_t = M_{t-1} + \eta \cdot \mathcal{R}_M(\boldsymbol{\lambda}). \tag{2}$$

[1]The tensor decomposition is implemented by `TensorLy` (https://github.com/tensorly/tensorly). Other implemented decomposition methods can also be used.

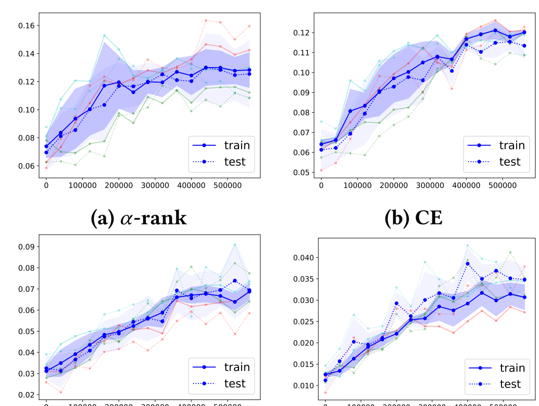


(a) α-rank (b) CE (c) FP (d) PRD

Figure 2: Results of RENES in simple case.

With the tensor decomposition, we can use a fixed size of action space of RENES, specified by r. The tensor decomposition can be viewed as a simple method of the abstraction [1, 2], and more sophisticated and decomposition methods can be considered in future works [3]. Then, RENES will optimize the modification of the games for multiple steps, e.g., 20, where the optimization process is formulated as a Markov Decision Process (MDP). We train the parameters in RENES with Proximal Policy Optimization [15].

4 EXPERIMENTS



(a) α-rank (b) CE (c) FP (d) PRD

Figure 3: Results of RENES in general case

In this section, we present the experimental results of RENES on large-scale normal-form games. We consider two cases: i) **simple case** where all games have the same size to verify the idea of modifying the games to boost the performance of inexact solvers, and ii) **general case** where the games have different sizes to verify that the design of RENES can handle the game with different sizes. Extensive experiments on large-scale normal-form games show that our method can further improve the approximation of NE of different solvers and can be generalized to unseen games.

REFERENCES

[1] Noam Brown and Tuomas Sandholm. 2015. Simultaneous abstraction and equilibrium finding in games. In *IJCAI*. 489–496.
[2] Noam Brown and Tuomas Sandholm. 2017. Safe and nested subgame solving for imperfect-information games. In *NeurIPS*. 689–699.
[3] Neil Burch, Michael Johanson, and Michael Bowling. 2014. Solving imperfect information games using decomposition. In *AAAI*.
[4] Xi Chen, Xiaotie Deng, and Shang-Hua Teng. 2009. Settling the complexity of computing two-player Nash equilibria. *J. ACM* 56, 3 (2009), 1–57.
[5] Constantinos Daskalakis, Paul W Goldberg, and Christos H Papadimitriou. 2009. The complexity of computing a Nash equilibrium. *SIAM J. Comput.* 39, 1 (2009), 195–259.
[6] Drew Fudenberg and Jean Tirole. 1991. *Game Theory*. MIT press.
[7] Paul W Goldberg, Christos H Papadimitriou, and Rahul Savani. 2013. The complexity of the homotopy method, equilibrium selection, and Lemke-Howson solutions. *ACM Transactions on Economics and Computation (TEAC)* 1, 2 (2013), 1–25.
[8] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).
[9] Tamara G Kolda and Brett W Bader. 2009. Tensor decompositions and applications. *SIAM Rev.* 51, 3 (2009), 455–500.
[10] Carlton E Lemke and Joseph T Howson, Jr. 1964. Equilibrium points of bimatrix games. *J. Soc. Indust. Appl. Math.* 12, 2 (1964), 413–423.
[11] Zhuwen Li, Qifeng Chen, and Vladlen Koltun. 2018. Combinatorial optimization with graph convolutional networks and guided tree search. In *NeurIPS*. 537–546.
[12] Roger B Myerson. 1999. Nash equilibrium and the history of economic theory. *Journal of Economic Literature* 37, 3 (1999), 1067–1082.
[13] John F Nash Jr. 1950. Equilibrium points in n-person games. *PNAS* 36, 1 (1950), 48–49.
[14] Shayegan Omidshafiei, Karl Tuyls, Wojciech M Czarnecki, Francisco C Santos, Mark Rowland, Jerome Connor, Daniel Hennes, Paul Muller, Julien Pérolat, Bart De Vylder, et al. 2020. Navigating the landscape of multiplayer games. *Nature Communications* 11, 1 (2020), 1–17.
[15] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).
[16] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *ICLR*.
[17] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. 2019. Graph transformer networks. In *NeurIPS*. 11983–11993.