Finding Optimal Nash Equilibria in Multiplayer Games via Correlation Plans

Extended Abstract

Youzhi Zhang
Centre for Artificial Intelligence and Robotics, Hong Kong Institute of Science & Innovation, CAS
Hong Kong SAR, China
youzhi.zhang@cair-cas.org.hk

Bo An
School of Computer Science and Engineering, Nanyang Technological University
Singnapore
boan@ntu.edu.sg

V. S. Subrahmanian
Department of Computer Science, Northwestern University
Evanston, USA
vss@northwestern.edu

ABSTRACT

Designing efficient algorithms to compute a Nash Equilibrium (NE) in multiplayer games is still an open challenge. In this paper, we focus on computing an NE that optimizes a given objective function. Finding an optimal NE in multiplayer games can be formulated as a mixed-integer bilinear program by introducing auxiliary variables to represent bilinear terms, leading to a huge number of bilinear terms, making it hard to solve. To overcome this challenge, we propose a novel algorithm called **CRM** based on a novel mixed-integer bilinear program with correlation plans for some subsets of players, which uses **C**orrelation plans with their **R**elations to strictly reduce the feasible solution space after the convex relaxation of bilinear terms while **M**inimizing the number of correlation plans to significantly reduce the number of bilinear terms. Experimental results show that our algorithm can be several orders of magnitude faster than the state-of-the-art baseline.

KEYWORDS

Nash equilibrium; Multiplayer game; Bilinear program

ACM Reference Format:
Youzhi Zhang, Bo An, and V. S. Subrahmanian. 2023. Finding Optimal Nash Equilibria in Multiplayer Games via Correlation Plans: Extended Abstract. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 3 pages.

1 INTRODUCTION

One of the important problems in artificial intelligence is the design of algorithms for agents to make decisions in interactive environments [13]. Designing efficient algorithms to compute NEs in multiplayer games is still an open challenge. In this paper, we focus on computing an optimal NE that optimizes a specific objective over the space of NEs. In the real world, we may need to optimize our objective over the space of NEs [3, 14]. Possible objectives could be maximizing social welfare (the sum of the players' expected utilities), maximizing the expected utilities of one player or several players, maximizing the minimum utility among players, minimizing the support sizes of the NE strategies, and so on. In addition, when there is a team of players in a game, team members need to consider finding an equilibrium that optimizes some objective [2, 4, 16–20]. Unfortunately, the problems mentioned above are NP-hard [3, 6]. In this paper, we propose a novel algorithm called CRM based on a novel mixed-integer bilinear program with correlation plans for some subsets of players.

Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom.

2 FINDING OPTIMAL NASH EQUILIBRIA

We consider a normal-form multiplayer game [15] with at least three players. The set of players as $N = \{1, \dots, n\}$; the set of all players' joint actions is $A = \times_{i \in N} A_i$, where A_i is the finite set of player i's pure strategies (actions) with $a_i \in A_i$; $u_i : A \to \mathbb{R}$ is player i's payoff function. $U_{max} = \max_{i \in N} \max_{a \in A} u_i(a)$, and $U_{min} = \min_{i \in N} \min_{a \in A} u_i(a)$. In addition, the set of (joint) mixed strategy profiles $X = \times_{i \in N} X_i$, where $X_i = \Delta(A_i)$ (i.e., the set of probability distributions over A_i) is the set of mixed strategies of player i. Let $-i$ be the set of all players excluding player i. x^* is a Nash Equilibrium (NE, and NEs for Nash Equilibria) [11] if, for each player i, x_i^* is a best response to x_{-i}^*, i.e., $u_i(x_i^*, x_{-i}^*) \geq u_i(x_i, x_{-i}^*), \forall x_i \in X_i$. The space of NEs in multiplayer games can be formulated as a multilinear program by directly extending the formulation for two-player games [14]:

$$u_i(a_i, x_{-i}) = \sum_{a_{-i} \in A_{-i}} u_i(a_i, a_{-i}) \prod_{j \in -i} x_j(a_{-i}(j)) \quad \forall i \in N, a_i \in A_i \tag{1a}$$

$$\textstyle\sum_{a_i \in A_i} x_i(a_i) = 1 \quad \forall i \in N \tag{1b}$$

$$1 - b_{a_i} \geq x_i(a_i) \quad \forall i \in N, a_i \in A_i \tag{1c}$$

$$u_i(x) \geq u_i(a_i, x_{-i}) \quad \forall i \in N, a_i \in A_i \tag{1d}$$

$$u_i(x) - u_i(a_i, x_{-i}) \leq b_{a_i}(U_{max} - U_{min}) \quad \forall i \in N, a_i \in A_i \tag{1e}$$

$$u_i(a_i, x_{-i}) \in [U_{min}, U_{max}], u_i(x) \in [U_{min}, U_{max}] \quad \forall i, a_i \tag{1f}$$

$$x_i(a_i) \in [0, 1], b_{a_i} \in \{0, 1\}, \quad \forall i \in N, a_i \in A_i, \tag{1g}$$

where we use the notations of utility functions $u_i(x)$ and $u_i(a_i, x_{-i})$ to represent the corresponding variables in the program. Eq.(1c) ensures that binary variable b_{a_i} is set to 0 when $x_i(a_i) > 0$ and can be set to 1 only when $x_i(a_i) = 0$; and Eq.(1e) ensures that the regret of action a_i equals 0 (i.e., $u_i(x) = u_i(a_i, x_{-i})$), unless $b_{a_i} = 1$ where the constraint $u_i(x) - u_i(a_i, x_{-i}) \leq (U_{max} - U_{min})$ always holds.

The multilinear program is usually transformed into a bilinear program to make the program solvable using global optimization solvers, e.g., Gurobi [9]. We use the binary tree definition to provide a recursive-binary definition for each subset of players in order to transform multilinear terms into bilinear terms. Our binary tree $T_{N'}$ of $N' \subseteq N$ with $|N'| \geq 2$ is that: 1) its root is N'; 2) its nodes are $\{N'' \mid N'' \subseteq N'\}$; 3) each of its leaf nodes is a singleton; and 4) each of its internal nodes N'' has two children N''_l and N''_r with $N''_l \cap N''_r = \emptyset$ and $N'' = N''_l \cup N''_r$, i.e., N'' is divided into two disjoint sets. Let $Ch(N'') = \{N''_l, N''_r\}$ be the set of N'''s children in $T_{N'}$, and $Ch(N'') = \emptyset$ if N'' is a singleton. Let $\mathcal{N}_{T_{N'}}$ be the set of internal nodes in $T_{N'}$. A **recursive-binary definition** of N' is a set of $Ch(N'')$ for all $N'' \in \mathcal{N}_{T_{N'}}$. Given a collection $\mathcal{N}$ of subsets of players, which is a subset of the power set of N, we say N' is **recursively-binarily defined** in $\mathcal{N}$ if there is a binary tree $T_{N'}$ such that all internal nodes in $T_{N'}$ are in $\mathcal{N}$, i.e., $\mathcal{N}_{T_{N'}} \subseteq \mathcal{N}$. $\mathcal{N}$ is a **binary collection** if each element N' in $\mathcal{N}$ is recursively-binarily defined in $\mathcal{N}$, and $\{-i \mid i \in N\} \subseteq \mathcal{N}$. The **vanilla binary collection** is $\overline{\mathcal{N}}$ that includes all of N's non-singleton proper subsets, where N''s children set $Ch(N')$ could be $\{N' \setminus \{j\}, \{j\} \mid j = \max_{i \in N'} i\}$ for each $N' \in \overline{\mathcal{N}}$.

To transform each multilinear term in Eq.(1a) into bilinear terms, given any binary collection $\mathcal{N}$ and each $N' \in \mathcal{N}$, we introduce auxiliary variable $P_{N'}(a_{N'}) \in [0,1]$ for each $a_{N'} \in A_{N'}$ with chains of bilinear equalities (i.e., Eq.(2b)) according to the definition of $Ch(N')$ of each $N' \in \mathcal{N}$. Specifically, $P_{N'}(a_{N'}) = x_i(a_i)$ for each singleton $N' = \{i\}$. Then we can transform multilinear constraints in Eq.(1a) into the following constraints with chains of bilinear equalities (i.e., Eq.(2b)):

$$u_i(a_i, x_{-i}) = \textstyle\sum_{a_{-i} \in A_{-i}} u_i(a_i, a_{-i}) P_{-i}(a_{-i}) \ \forall i \in N, a_i \in A_i \tag{2a}$$

$$P_{N'}(a_{N'}) = P_{N'_l}(a_{N'_l}) P_{N'_r}(a_{N'_r}) \quad \forall N' \in \mathcal{N}, Ch(N') = \{N'_l, N'_r\}, \quad a_{N'} = (a_{N'_l}, a_{N'_r}) \in A_{N'}. \tag{2b}$$

$$P_{N'}(a_{N'}) \in [0,1]. \quad \forall a_{N'} \in A_{N'}, N' \in \mathcal{N}, \tag{2c}$$

After the transformation, Eqs.(1b)-(1g) and (2) represent the space of NEs. An optimal NE is an NE optimizing an objective. Then finding an optimal NE requires optimizing an objective function $g(x)$ over this space of NEs:

$$\max_x g(x) \tag{3a}$$

$$\text{s.t. Eqs.}(1b) - (1g), (2). \tag{3b}$$

An important step used by state-of-the-art algorithms to solve such bilinear programs is to use convex relaxation to replace each bilinear term in the program [7, 8], which significantly enlarges the feasible solution space. To reduce this space, we propose to exploit correlation plans with their relations. For each N' in any binary collection $\mathcal{N}$, a correlation plan of N' is a probability distribution $P_{N'}$ over $A_{N'}$ (i.e., $P_{N'} \in \Delta(A_{N'})$), which satisfies:

$$\textstyle\sum_{a_{N'} \in A_{N'}} P_{N'}(a_{N'}) = 1 \quad \forall N' \in \mathcal{N}. \tag{4}$$

We now exploit relations between correlation plans for elements in $\mathcal{N} \cup \{\{i\} \mid i \in N\}$ according to the binary definition.

$$\sum_{a_{N'} \in A_{N'}, a_{N'}(i) = a_i} P_{N'}(a_{N'}) = x_i(a_i) \quad \forall i \in N', a_i \in A_i, N' \in \mathcal{N} \tag{5a}$$

$$\sum_{a_{N'} = (a_{N'_l}, a_{N'_r}) \in A_{N'}} P_{N'}(a_{N'}) = P_{N'_l}(a_{N'_l}) \quad \forall a_{N'_l} \in A_{N'_l}, |N'_l| > 1 \tag{5b}$$

$$\sum_{a_{N'} = (a_{N'_l}, a_{N'_r}) \in A_{N'}} P_{N'}(a_{N'}) = P_{N'_r}(a_{N'_r}) \quad \forall a_{N'_r} \in A_{N'_r}, |N'_r| > 1, \tag{5c}$$

where conditions $|N'_l| > 1$ and $|N'_r| > 1$ ensure Eq.(5a) and Eq.(5) do not generate the same constraints, and $CH(N') = \{N'_l, N'_r\}$.

Now we explicitly restrict the feasible solution space by adding Eqs.(4) and (5) to Program (3) for any binary collection $\mathcal{N}$:

$$\max_x g(x) \tag{6a}$$

$$\text{s.t. Eqs.}(1b) - (1g), (2), (4), (5). \tag{6b}$$

Algorithm 1 Generate $\underline{\mathcal{N}}$

1: Build a full binary tree T_{-n} with the height $\lceil \log_2(n-1) \rceil$ for $-n$ with the set of internal nodes $\mathcal{N}_{T_{-n}}$ and $|\mathcal{N}_{T_{-n}}| = n - 2$
2: **for** each i in $\{1, \ldots, n-1\}$ **do**
3: Search T_{-n} to replace i with n in each node including i to form a binary tree T_{-i} with the set of internal nodes $\mathcal{N}_{T_{-i}}$
4: **end for**
5: $\underline{\mathcal{N}} \leftarrow \cup_{i \in N} \mathcal{N}_{T_{-i}}$.

Table 1: Results. "∞": no solutions are returned. The last three games are six-player three-action GAMUT games.

	Runtime (Percentage of Games not Solved) (Utility Gap)			
(n, m)	CRM	MIBP	ENUMPOLY	EXCLUSION
(3, 2)	**0.01**	0.02	0.03	31 (gap:15%)
(7, 2)	**25**	42 (20%)	1000 (97%)	835 (80%) (gap:53%)
(3, 5)	**0.2**	0.3	1000 (100%)	1000 (100%) (gap:67%)
(3, 13)	**38**	342 (27%)	1000 (100%)	1000 (100%) (gap:∞)
Collaboration	**1**	967 (97%)	1000 (100%)	1000 (100%) (gap:81%)
Random LEG	**2**	1000 (100%)	1000 (100%)	986 (97%) (gap:11%)
Uniform LEG	**2.2**	1000 (100%)	1000 (100%)	986 (97%) (gap:11%)

Theorem 1. *The optimal solution of Program (6) maximizes $g(x)$ over the space of NEs.*

It is straightforward to use $\overline{\mathcal{N}}$ in Program (6), where we need to add a set of linear constraints and bilinear constraints for each correlation plan corresponding to each element in any binary collection $\mathcal{N}$. However, $\overline{\mathcal{N}}$ is too large. To reduce the number of bilinear terms, we propose building minimum-height binary trees to obtain a new binary collection with the minimum size, which gives us the minimum set of correlation plans. This procedure is shown in Algorithm 1 generating $\underline{\mathcal{N}}$. Our algorithm, CRM, is solving Program (6) based on $\underline{\mathcal{N}}$, i.e., $\mathcal{N}$ in Eqs.(2), (4), and (5) is replaced by $\underline{\mathcal{N}}$.

Theorem 2. *$\underline{\mathcal{N}}$ generated by Algorithm 1 is a binary collection, and $O(n \log n)$ for the size of $\underline{\mathcal{N}}$ is the minimum size of all binary collections of a game.*

3 EXPERIMENTS

We evaluate our approach on randomly generated games with n players and m actions for each player and GAMUT [12] games. We compare our CRM to the state-of-the-art baselines: 1) **MIBP** [5, 14]: the equivalent of solving Program (3) based on $\overline{\mathcal{N}}$; 2) **EXCLUSION** [1]: the first implemented algorithm guarantees to converge to an NE (that may not be optimal, and then we measure the utility gap); and 3) **ENUMPOLY** [10]: an algorithm in Gambit, which tries to find all NEs, which can then choose an optimal NE from the output of all NEs. CRM and MIBP can guarantee finding an optimal NE. The objective function maximizes the expected utility of player n. We use the non-convex solver of Gurobi 9.5 to solve all mixed-integer bilinear programs with the optimality gap 0.0001. EXCLUSION uses this optimality gap as well, which is significantly smaller than 0.001 in [1]. We set a time limit of 1000 seconds for each case and measure the percentage of games that are not solved within the time limit. Table 1 shows the results average on 30 cases, where CRM can be two or three orders of magnitude faster than baselines.

ACKNOWLEDGMENTS

This research is supported by the InnoHK Fund. Some of the experiments were facilitated by ONR grant N00014-20-1-2407.

REFERENCES

[1] Kimmo Berg and Tuomas Sandholm. 2017. Exclusion method for finding Nash equilibrium in multiplayer games. In *AAAI*. 383–389.

[2] Andrea Celli and Nicola Gatti. 2018. Computational results for extensive-form adversarial team games. In *AAAI*. 965–972.

[3] Vincent Conitzer and Tuomas Sandholm. 2003. Complexity results about Nash equilibria. In *IJCAI*. 765–771.

[4] Gabriele Farina, Andrea Celli, Nicola Gatti, and Tuomas Sandholm. 2021. Connecting optimal ex-ante collusion in teams to extensive-form correlation: Faster algorithms and positive complexity results. In *ICML*. PMLR, 3164–3173.

[5] Sam Ganzfried. 2021. Fast complete algorithm for multiplayer Nash equilibrium. *arXiv preprint arXiv:2002.04734* (2021).

[6] Itzhak Gilboa and Eitan Zemel. 1989. Nash and correlated equilibria: Some complexity considerations. *Games and Economic Behavior* 1, 1 (1989), 80–93.

[7] Ambros M Gleixner, Timo Berthold, Benjamin Müller, and Stefan Weltge. 2017. Three enhancements for optimization-based bound tightening. *Journal of Global Optimization* 67, 4 (2017), 731–757.

[8] Gurobi. 2020. Non-Convex Quadratic Optimization. https://pages.gurobi.com/rs/181-ZYS-005/images/2020-01-14_Non%20Convex%20Quadratic%20Optimization%20in%20Gurobi%209.0%20Webinar.pdf. Accessed: 2021-09-20.

[9] Gurobi. 2021. Gurobi Optimizer Reference Manual. https://www.gurobi.com/documentation/9.1/refman/index.html. Accessed: 2021-09-20.

[10] Richard D. McKelvey, Andrew M. McLennan, and Theodore L. Turocy. 2014. Gambit: Software tools for game theory, version 16.0.1. *http://www.gambit-project.org* (2014).

[11] John Nash. 1951. Non-cooperative games. *Annals of Mathematics* (1951), 286–295.

[12] Eugene Nudelman, Jennifer Wortman, Yoav Shoham, and Kevin Leyton-Brown. 2004. Run the GAMUT: A comprehensive approach to evaluating game-theoretic algorithms. In *AAMAS*, Vol. 4. 880–887.

[13] Stuart J Russell and Peter Norvig. 2016. *Artificial Intelligence: A Modern Approach*. Malaysia; Pearson Education Limited.

[14] Tuomas Sandholm, Andrew Gilpin, and Vincent Conitzer. 2005. Mixed-integer programming methods for finding Nash equilibria. In *AAAI*. 495–501.

[15] Yoav Shoham and Kevin Leyton-Brown. 2008. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press.

[16] Bernhard von Stengel and Daphne Koller. 1997. Team-maxmin equilibria. *Games and Economic Behavior* 21, 1-2 (1997), 309–321.

[17] Youzhi Zhang and Bo An. 2020. Computing team-maxmin equilibria in zero-sum multiplayer extensive-form games. In *AAAI*. 2318–2325.

[18] Youzhi Zhang and Bo An. 2020. Converging to team-maxmin equilibria in zero-sum multiplayer games. In *ICML*. 11033–11043.

[19] Youzhi Zhang, Bo An, and Jakub Černỳ. 2021. Computing ex ante coordinated team-maxmin equilibria in zero-sum multiplayer extensive-form games. In *AAAI*, Vol. 35. 5813–5821.

[20] Youzhi Zhang, Bo An, and V.S. Subrahmanian. 2022. Correlation-based algorithm for team-maxmin equilibrium in multiplayer extensive-form games. In *IJCAI*. 606–612.