Policy Abstraction and Nash Refinement in Tree-Exploiting PSRO

Christine Konicki*
Michigan Tech Research Institute
Ann Arbor, USA
ckonicki@mtu.edu

Mithun Chakraborty
University of Michigan
Ann Arbor, USA
dcsmc@umich.edu

Michael P. Wellman
University of Michigan
Ann Arbor, USA
wellman@umich.edu

ABSTRACT

Policy Space Response Oracles (PSRO) interleaves empirical game-theoretic analysis with deep reinforcement learning (DRL) to solve games too complex for traditional analytic methods. Tree-exploiting PSRO (TE-PSRO) is a variant of this approach that iteratively builds a coarsened empirical game model *in extensive form* using data obtained from querying a simulator that represents a detailed description of the game. We make two main methodological advances to TE-PSRO that enhance its applicability to complex games of imperfect information. First, we introduce a scalable representation for the empirical game tree where edges correspond to *implicit policies* learned through DRL. These policies cover conditions in the underlying game abstracted in the game model, supporting sustainable growth of the tree over epochs. Second, we leverage extensive form in the empirical model by employing refined Nash equilibria to direct strategy exploration. To enable this, we give a modular and scalable algorithm based on generalized backward induction for computing a subgame perfect equilibrium (SPE) in an imperfect-information game. We experimentally evaluate our approach on a suite of games including an alternating-offer bargaining game with outside offers; our results demonstrate that TE-PSRO converges toward equilibrium faster when new strategies are generated based on SPE rather than Nash equilibrium, and with reasonable time/memory requirements for the growing empirical model.

KEYWORDS

Game Theory, Extensive-form Games, PSRO

ACM Reference Format:
Christine Konicki*, Mithun Chakraborty, and Michael P. Wellman. 2025. Policy Abstraction and Nash Refinement in Tree-Exploiting PSRO. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 9 pages.

1 INTRODUCTION

Empirical game-theoretic analysis (EGTA) [16, 18] reasons about complex game scenarios through ***empirical game*** models estimated from simulation data. A popular form of EGTA is the ***policy space response oracles*** (PSRO) framework [9] (Fig. 1), in which the empirical game is iteratively extended by adding best responses (BRs) derived from ***deep reinforcement learning*** (DRL). The vast majority of prior work on EGTA and PSRO [1, 19] represents the empirical game in normal form even though the real underlying game consists of agents' strategies interacting via sequential decisions under various unknowns. McAleer et al. [12] introduced ***XDO*** as an alternative to PSRO that maintains an empirical game in extensive form, to capture a combinatorial space of strategies with the choice of actions at each decision point in the game tree. We originally proposed and evaluated a ***tree-exploiting*** version of EGTA (TE-EGTA) that maintains an empirical game in an extensive form based on a coarsening of the underlying game [8]. This work demonstrated that a significant improvement in model accuracy and strategy exploration, compared to normal-form EGTA, can be achieved by using the tree structure to model even a little of the information-revelation and action-conditioning patterns of the underlying game.

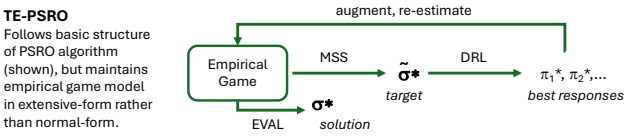

Figure 1: Basic PSRO loop. In each iteration (or *epoch*), an empirical game model is extended, based on best responses (BRs) to target profile $\tilde{\sigma}^*$ derived from the current empirical game by the solver MSS. The BRs are computed using deep RL applied to the game simulator. EVAL is a solver (not necessarily the MSS) applied to a model to assess its quality.

A key step in PSRO is augmenting the empirical game model with new BR results. This is straightforward for a normal-form model: add the new strategies to the game matrix, and estimate payoffs for new profiles using simulation. For PSRO using an extensive-form model, where we can no longer treat a BR as an atomic entity, we face new questions such as the following. In what precise sense does the empirical game tree *coarsen* or *abstract* the underlying multiagent scenario? Relatedly, how should we incorporate elements of the BRs (detailed policy specifications) at appropriate places in the empirical game tree? In our previous approach towards tree-exploiting PSRO (TE-PSRO) [8], we addressed these challenges by systematically coarsening away select (non-strategic) stochastic events in the underlying game. This approach is not general or scalable in the sense that it relies on the use of stochastic events to model imperfect information in the underlying game. In this paper, we reformulate TE-PSRO by developing a method to abstract broad swaths of the *state* and *observation* spaces of the underlying game, providing a more implicit rendering of games with complex information structure, including high degrees of imperfect information. We also introduce other methodological advances that enhance the power of TE-PSRO in many directions.

*Konicki worked on this paper while a PhD student at the University of Michigan.

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

First, to address the abstraction issue, we use two distinct formulations for the game of interest. The *underlying game as represented by the simulator* is defined in terms of a state space, agent actions, and the observations, successor states, and rewards (stochastically) resulting from applying actions in a given state. This formulation is the natural one for DRL algorithms, which can interact directly with the simulator. On the other hand, the *empirical game model* employs an extensive-form tree representation, which is the natural object of game-solving algorithms. To bridge the two formulations, the edges in the empirical game tree correspond to ***abstract policies*** executable in the simulator. These abstract policies, derived through DRL, map agent observations to actions.[1] The empirical game includes only select elements of this history, rendering much of the simulator state space and observation space implicit in this abstracted formulation. At what level to incorporate observation details in the game model is a design choice, entailing tradeoffs in computation and fidelity.

Second, we face additional computational tradeoffs regarding how much to elaborate the game model based on DRL computations. In each iteration (or *epoch*), PSRO solves the current empirical game using a ***meta-strategy solver*** (MSS). It then derives an approximate best response for each player using DRL, assuming the other players follow the latest MSS solution. A straightforward approach would apply the derived response throughout the empirical game [8, 12]. This, however, could lead the game tree to grow at an exponential rate, severely limiting the feasible number of PSRO epochs. We propose to control this growth by adding the new DRL policies to only a select set of information sets in the empirical game.

A final question we address regards the choice of MSS for TE-PSRO. Previous research has considered a range of MSSs that operate on normal-form games, and observed a significant impact on PSRO efficacy [9, 17]. We now have the opportunity to consider new MSSs that exploit the tree structure of an empirical game in extensive form, in particular, refined Nash equilibria.

To illustrate and evaluate our upgraded TE-PSRO approach, we employ two (perfect-recall) games with significantly different imperfect information structures that non-trivially extend stylized games from the literature. In our first game that we call Bargain, two players with private valuations over a set of indivisible items negotiate how to split the set up between them through an alternating-offer protocol. The scenario features imperfect information about the other party's valuation as well as choices for signaling regarding the value of a private *outside option* that each player has recourse to in the event of negotiation failure. This sequential bargaining game extends a well-known two-party multi-issue negotiation task [3, 4, 10, 11].

The second game is a general-sum abstraction of the card game Goofspiel [5] that we call GenGoof. As a clean model of multi-round multiagent interactions with considerable strategic depth, Goofspiel has been extensively analyzed in the game-theory literature, more recently serving as a testbed for game-playing AI algorithms [2]. GenGoof proceeds over an arbitrary number of rounds, each including a discrete stochastic event (defined over a support diminishing every round by the single realized outcome) followed by all players choosing one discrete action each (effectively simultaneously); the payoff of each player at game termination is the sum of arbitrary per-round rewards.

Our key contributions are:

- A general scheme for abstract policy representation that supports flexible implementation of TE-PSRO for complex games of imperfect information (§4.1 and §4.2).
- An approach to control the growth of empirical game trees through selective incorporation of best responses at particular information sets (§4.3 and App. B.1).
- A new algorithm that computes a subgame perfect equilibrium (SPE) of an imperfect-information game (§5).
- Experimental demonstration of the efficacy of SPE over NE as MSS for TE-PSRO on a variety of complex sequential games of imperfect information (§3 and §6); our experiments address three aspects of the complete TE-PSRO loop: the effectiveness of our augmentation heuristic in controlling the empirical game growth rate, the power of our SPE computation algorithm, and a comparison of MSS choices including refined equilibria which are feasible only for extensive-form empirical games.

All appendices referenced below are available in the full version of the paper at https://arxiv.org/abs/2502.02901. The code base used for our experiments is available at https://github.com/ckonicki-umich/AAMAS25.

2 TECHNICAL PRELIMINARIES

An ***extensive-form game*** (EFG) is a tuple

$$G = \langle\, N,\, H,\, V,\, \{\mathcal{I}_j\}_{j=0}^{n},\, \{A_j\}_{j=1}^{n},\, X,\, P,\, u \,\rangle,$$

where $N = \{0, \ldots, n\}$ is the set of players or agents; H is a finite tree of histories divided into subsets of ***terminal nodes*** or leaves Z and decision nodes D; V is a function assigning each decision node h to an acting player; $\mathcal{A}_j(\cdot)$ is the set of actions available at each decision node; u is a function mapping each $z \in Z$ to a ***utility*** vector $\{u_j(z)\}_{j=1}^{n}$; in games of imperfect information, the set $\mathcal{I}_j$ is a partition of $V^{-1}(j)$ where each $I \in \mathcal{I}_j$ is an ***information set*** (or ***infoset***) of j. All nodes $h \in I$ are indistinguishable to player j, meaning their action spaces are also indistinguishable and denoted $A_j(I)$. The directed edge connecting any $h \in I$ to its child represents a transition resulting from $V(h)$'s move. We assume ***perfect recall*** [15, Definition 5.2.3]. A node h where $V(h) = 0$ is called a ***chance node*** controlled by Nature, with a set of possible outcomes $X(h)$ and probability distribution $P(\cdot \mid h)$ over $X(h)$.

Since the underlying or ***true game*** corresponding to the simulator is too large to be represented directly with a tree, we instead express it in a state-action formulation. A play of the game is a sequence of actions taken by the players (including Nature), where each action leads to a ***world state*** $w \in \mathcal{W}$. The joint space of actions is given by $\mathcal{A} = \bigotimes_{j=1}^{n} \mathcal{A}_j$, and the set of legal actions for agent j at world state w is given by $\mathcal{A}_j(w) \subseteq \mathcal{A}_j$. The probability distribution of the world state w' following joint action $a = (a_1, \ldots, a_n) \in \mathcal{A}$ taken in world state w is given by a transition function $\mathcal{T}(w, a) \in \Delta^{\mathcal{W}}$. Upon transitioning to w' from w

[1] McAleer et al. [12] define a variant of XDO called Neural XDO that likewise employs policies represented as neural networks. Rather than incorporate these as abstract policies in an empirical game tree, Neural XDO instead relies on methods like neural fictitious self-play [6] that perform game analysis directly in the space of neural network policies.

via a, agent j makes a partial ***observation*** $o_j = O_j(w, a, w')$ instead of fully observing w'. A reward $\mathcal{R}_j(w)$ is given to agent j at each $w \in \mathcal{W}$, and the game ends when a ***terminal world state*** is reached. In this formulation, a history at time t is the sequence of world states and actions given by $h = (w^1, a^1, \ldots, w^t)$; histories $z \in Z$ where w^t is a terminal world state are terminal histories. It follows that $R_j(h)$ and $\mathcal{A}_j(h)$ are the reward and action space for agent j in the last world state of history h. An ***information state*** (or ***infostate***) for agent j, denoted I_j, is a sequence of agent j's observations and actions up to a point t in the game, given by $I_j(h) = (a_j^1, o_j^2, a_j^2, \ldots, o_j^t) \equiv I_j$. Since agent j cannot distinguish between the histories of $I_j(h)$, it follows that $\mathcal{A}_j(I_j(h)) = \mathcal{A}_j(h)$. The complete set of infostates for player j is again given by $\mathcal{I}_j$. We use hatted symbols to denote components of the empirical game tree (e.g., $\hat{\mathcal{I}}_j$ for infosets) to distinguish them from analogous components of the true game.

A ***pure strategy*** for player j specifies the action that j selects at each information set. A ***mixed strategy*** σ_j defines a probability distribution over the action space at each of j's information sets. A ***strategy profile*** is given by $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_n)$, and $\boldsymbol{\sigma}_{-j}$ denotes the strategies of all players other than j. Σ_j is the set of all strategies available to player j, and $\Sigma = \bigtimes_{j=1}^n \Sigma_j$ denotes the space of joint strategy profiles. A terminal history z is reached by $\boldsymbol{\sigma}$ with a ***reach probability*** $r(z, \boldsymbol{\sigma}) = \prod_{j \in N} r_j(z, \sigma_j)$ where $r_j(z, \sigma_j)$ is the probability that player j chooses actions that lead to z, including Nature's contribution $r_0(z)$. The ***payoff*** of $\boldsymbol{\sigma}$ to player j is given by its expected utility $U_j(\boldsymbol{\sigma})$. Player j's ***regret*** from playing σ_j as part of $\boldsymbol{\sigma}$ is given by $\mathrm{Reg}_j(\boldsymbol{\sigma}) = \max_{\sigma \in \Sigma_j} U_j(\sigma, \boldsymbol{\sigma}_{-j}) - U_j(\boldsymbol{\sigma})$. The profile regret of $\boldsymbol{\sigma}$ is the sum of player regrets: $\mathrm{Reg}(\boldsymbol{\sigma}) = \sum_{j=1}^n \mathrm{Reg}_j(\boldsymbol{\sigma})$. A strategy profile $\boldsymbol{\sigma}$ with $\mathrm{Reg}(\boldsymbol{\sigma}) = 0$ is a ***Nash equilibrium*** (NE).

3 DESCRIPTION OF GAMES STUDIED

We will now describe in detail the two games used in our experimental assessment of TE-PSRO in §6. The game analyst using TE-PSRO has no direct access to a game description at this level of detail, but can query a simulator based on such a description for data samples relevant to game histories induced by input strategy profiles.

3.1 Bargain

In this game, two players negotiate the division of m discrete items of τ types. We represent the item pool by a vector $\mathbf{p}$ where the i^{th} entry p_i, $i \in \{1, \ldots, \tau\} \equiv [\tau]$, is the number of items of type i; $\sum_{i=1}^{\tau} p_i = m$. Each player $j \in \{1, 2\}$ has a private valuation over the items given by a vector $\mathbf{v}_j$ of non-negative integers such that the i^{th} entry $v_{j,i}$ is player j's value for one item of type i. In each game instance, $(\mathbf{v}_1, \mathbf{v}_2)$ are sampled uniformly at random from the collection $\mathcal{V}$ of all vector pairs satisfying three constraints. First, for both players, the total value of all items is a constant: $\mathbf{v}_j \cdot \mathbf{p} = \bar{V}$, $j \in \{1, 2\}$. Second, each item type must have nonzero value for at least one player: $\forall i \in [\tau].v_{1,i} + v_{2,i} > 0$. Finally, some item type must have nonzero value for both players: $\exists i \in [\tau].v_{1,i}v_{2,i} > 0$.

An additional feature of our game is that each player j has a private ***outside offer*** in the form of a vector of items $\mathbf{o}_j$, defining the fallback payoff the player obtains if no deal is reached. This offer is drawn from a distribution $P_j(\cdot)$ at the start of each game instance. During negotiation, a player j may choose to reveal coarsened information about its outside offer to the other player in the form of a binary signal which is L (resp. H) if the value of the offer $\mathbf{o}_j \cdot \mathbf{v}_j$ is at most (resp. greater than) a fixed threshold ν where $1 < \nu < \bar{V}$.

In each of a finite number $T > 0$ of negotiation rounds, the players take turns proposing a partition of the pool between themselves, with player 1 moving first in each round. In its turn, a player can accept the latest offer from the other player (DEAL), end negotiations (WALK), or make an offer-revelation combination of the form (ω, R). Offer $\omega \in \{(\mathbf{p}_1, \mathbf{p}_2) \mid \mathbf{p}_1 + \mathbf{p}_2 = \mathbf{p}\}$ is a proposed partition of the items, with $\mathbf{p}_j$ a vector of τ non-negative integers representing player j's share. Revelation $R \in \{\text{TRUE}, \text{FALSE}\}$ represents that player's decision to either disclose its signal (TRUE) in that turn or not (FALSE). We also include a discount factor $\gamma \in (0, 1]$ to capture preference for reaching deals sooner. Negotiation fails if a player chooses WALK in any round $\rho \in \{1, \ldots, T\}$ or T rounds pass without any player choosing DEAL. In case of failure in round ρ, each player j receives a reward of $\gamma^\rho \mathbf{o}_j \cdot \mathbf{v}_j$ from its outside offer. If a proposed partition $(\mathbf{p}_1, \mathbf{p}_2)$ is accepted in round ρ, then the reward to j is $\gamma^{\rho-1}\mathbf{p}_j \cdot \mathbf{v}_j$.

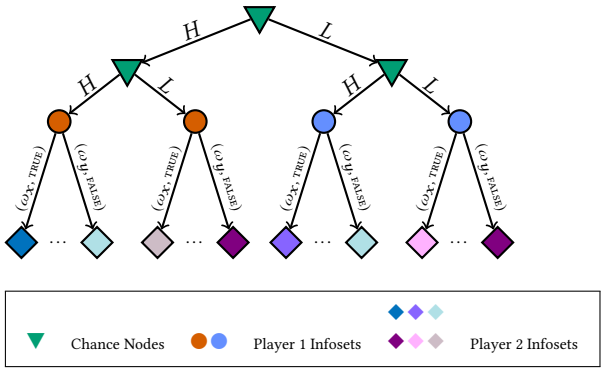

Figure 2: Part of game tree llustrating the effect of Player 1's decision R on Player 2's infoset structure.

Fig. 2 displays a partial extensive-form representation of a simulation of Bargain after each player's valuation has already been sampled. At the first two levels, the simulator samples outside offer signals for the two players from $\{H, L\}$; if L (resp. H) is drawn for player j, its actual outside offer is sampled uniformly at random from all possible item vectors $\mathbf{o}$ of cumulative value $\mathbf{o} \cdot \mathbf{v}_j$ above 0 and at most the threshold ν (resp. above ν and at most $\bar{V}$). Thus, setting P_j reduces to picking a probability of the signal being H for player j. Next, player 1 chooses an action, comprising an offer ω and revelation R. Player 2 now has four distinguishable histories that result when player 1 reveals its signal and two non-singleton information sets that result when player 1 chooses not to reveal. The game continues (not shown) with the action of player 2, and alternation between the players for another $T - 1$ rounds.

3.2 GenGoof

GenGoof is parametrized by a positive integer K which determines the number of game rounds $(K - 1)$ as well as the size of each player's action space (K).

First, a discrete stochastic event with K outcomes occurs at the game root, that is, $|X(\emptyset)| = K$. Let e_1 denote the realized outcome

of this event. Player 1 observes e_1 and chooses one of K actions, say a_1^k, from $\mathcal{A}_1(I(e_1)) = \{a_1^k\}_{k=1}^K$. Then, player 2 observes e_1 but not player 1's action, say a_2^k, and also chooses one of K actions from $\mathcal{A}_2(I(e_1 a_1^k)) = \{a_2^k\}_{k=1}^K$, ending round 1. Round 2 begins with the realization e_2 of a second stochastic event with support $X\left(e_1 a_1^k a_2^k\right) = X(\emptyset) \setminus \{e_1\}$. Player 1 then observes the history up to and including e_2 before choosing one of K actions, followed by player 2 who observes all but player 1's second chosen action. This process repeats until the final round $K-1$ where a stochastic event with only 2 possible outcomes occurs, followed by player 1 and player 2 both observing the history until the final stochastic event realization and picking one of K actions each, ending the game.

To complete the game description, we define the probability distribution of each stochastic event and each leaf utility as follows (these are also game parameters that are hidden from the game analyst). For each instance of GenGoof, we sample a single K-outcome categorical probability distribution uniformly at random for the stochastic event in round 1; for $k \in \{2, 3, \ldots, K-1\}$, the distribution of the round-k stochastic event is obtained by renormalizing the round-$(k-1)$ over the residual support after eliminating the outcome realized in round $(k-1)$. For example, the probability distribution of the round-2 stochastic event given that e_1 occurred in round 1 is

$$P\left(e_2 \mid e_1 a_1^k a_2^k\right) = \frac{P(e_2 \mid \emptyset)}{\sum_{e' \in X(\emptyset)\setminus\{e_1\}} P(e' \mid \emptyset)} \quad \forall e_2 \in X(\emptyset) \setminus \{e_1\}.$$

For each possible combination of the stochastic event outcome and the two players' action choices in each round of any game instance, we choose a reward for each player uniformly at random from $[0, u_{\max}]$ for a positive real number $u_{\max}$; we set the leaf utility for each player equal to the sum of the player's rewards over all $K-1$ rounds in the corresponding history. Thus, for every leaf $z \in Z$ and player $j \in \{1, 2\}$, $u_j(z) \sim \mathbb{U}([0, u_{\max}(K-1)])$ where $\mathbb{U}(\mathcal{S})$ denotes the uniform distribution over the set $\mathcal{S}$. Figure 7 in App. A illustrates the first round of gameplay in a particular instance of GenGoof.

4 TREE-EXPLOITING PSRO

Our domain of interest comprises game instances where expanding the full game out as an extensive-form tree for the purpose of game analysis is infeasible. TE-PSRO tackles this challenge by maintaining a coarsened and abstracted (yet extensive-form) version as its empirical game model. The full game is specified in the form of a gameplay simulator, which is formalized in terms of the world state framework (§2). A key question for TE-PSRO (Fig. 1) is how to translate new best-response results into elements that can be systematically incorporated into an abstracted empirical game model as part of the model augmentation operation. Our approach bridges the detailed state-space model of the simulator and the simplified game tree using the concept of abstract policies.

4.1 Abstract Policies

In the underlying game, the space of possible infostates of player j is given by $\mathcal{I}_j$, and a policy π_j specifies j's action $a \in \mathcal{A}_j(I)$ for each $I \in \mathcal{I}_j$. We represent such policies in our implementation by neural networks, encoded as a set of weights for a given architecture. In the empirical game $\hat{G}$, player j's information possibilities are described by its infosets $\hat{\mathcal{I}}_j$. In general, there need be no particular structural relationship between $\mathcal{I}_j$ and $\hat{\mathcal{I}}_j$, though typically they will both be defined in terms of a shared set of primitive observations. We capture the connection by a function $\phi : \mathcal{I}_j \to \hat{\mathcal{I}}_j$, where $\phi(I_j)$ is the empirical game infoset corresponding to underlying infostate I_j.

Given the distinct formulations, policies π_j are executable in the simulator, but cannot be directly interpreted within the framework of empirical game $\hat{G}$. Nevertheless, we can incorporate them as actions in $\hat{G}$. For a given policy π_j^x, we treat the label "π_j^x" as a potentially allowable action for any infoset $\hat{I}_j$. From the empirical game perspective, "π_j^x" is an abstract policy. An overall game-tree strategy specifies an action for every infoset in $\hat{\mathcal{I}}_j$. To execute a game-tree strategy profile in the simulator, we simply trace through the tree, applying selected abstract policies at each infoset. The selected abstract policy remains in force for player j until a new infoset is reached where it is j's turn to move. Though uninterpreted in the game tree itself, these abstract policies have full access to the information state from the simulation needed for execution.

4.2 Best Response: Deep Reinforcement Learning

With the ability to simulate profiles over the empirical game strategy space, we can employ the simulator within a deep RL algorithm to derive best responses π_j^*. Our implementation employs the DQN algorithm [13], which combines a feed-forward neural network parameterized by ϑ with temporal difference learning and a second target network to estimate Q-values over time given $I \in \mathcal{I}_j$.

As an illustration, we provide details of our deep Q-network for Bargain. The input to the neural network representing π_j^* for this game is an encoding of player j's current information state I. $\lceil \log_2(\bar{V}) \rceil$ bits are allotted for player j's valuation $\mathbf{v}_j[i]$, for each $i \in [\tau]$. One bit is allotted for player j's outside offer signal. For each player's turn in the game, $\mathbf{p}[i] + 1$ bits are allotted per item type $i \in [\tau]$ to represent a partition of $\mathbf{p}$, plus one bit for the decision to reveal the signal or not. Two bits are allotted to represent the other player's signal: 00 means no reveal so far, 01 means L, and 10 means H. One final bit is allotted to be set to 1 when negotiations are complete. The output of the network is an $|\mathcal{A}_j|$-long vector containing the Q-values of each action in $\mathcal{A}_j$ given the input infostate vector. Our parameter settings, optimized via hyperparameter tuning, are included in App. G. After successfully training player j's DQN, the learned weights of π_ϑ are saved and mapped to the abstract policy label "π_j^x" in $\hat{G}$ (§4.1).

4.3 Augmenting the Empirical Game Model

Given BRs π^* computed from DRL, the next step is to augment the empirical game $\hat{G}$ (Fig. 1). Though an abstract policy is potentially applicable at any point in the game tree, adding π_j^* to every $I \in \hat{\mathcal{I}}_j$ could lead to unsustainable growth in $\hat{G}$.

Our approach is to select a fixed number M of infosets to augment for each player. Our selection is based on an assessment of the ***gain*** Γ of playing π_j^* instead of $\tilde{\boldsymbol{\sigma}}_j^*$ at candidate infosets $I \in \hat{\mathcal{I}}_j$

where $\tilde{\sigma}^*$ is the BR target at the current TE-PSRO epoch. Recall (§2) that a terminal history in the underlying game is expressed as a sequence of world states and actions: $z = (w^1, a^1, \ldots, w^t)$, and that $R_j(z)$ is the associated reward for player j. The reach probability of z given that history h of length t_h was reached is

$$r(z \mid h, \hat{\sigma}) = \prod_{\ell=t_h}^{t} \hat{\sigma}(\phi(w^\ell))(\pi^x) \cdot \mathbb{1}\{\pi^x(w^\ell) = a^\ell\} \cdot T(w^\ell, a^\ell)(w^{\ell+1}).$$

The expected payoff to j for playing $\hat{\sigma}_j$ in response to $\tilde{\sigma}^*_{-j}$ at $I \in \hat{\mathcal{I}}_j$ is given by

$$U_j(\hat{\sigma}_j, \tilde{\sigma}^*_{-j}, I) = \sum_{h \in H \mid \phi(h) \in I} \sum_{z \in Z} r(z \mid h, \hat{\sigma}_j, \tilde{\sigma}^*_{-j}) R_j(z).$$

Let $\hat{\sigma}|_{I \to \pi^*_j}$ denote a strategy profile identical to $\hat{\sigma}$ except that player j selects π^*_j at I. Gain K is equal to the product of the gain to player j of $\tilde{\sigma}^*|_{I \to \pi^*_j}$, given that I is reached, and the probability of reaching the set of histories in the underlying game that translate into I:

$$\Gamma = r(I, \tilde{\sigma}^*) \cdot \left(U_j(\, \tilde{\sigma}^*|_{I \to \pi^*_j}, I) - U_j(\tilde{\sigma}^*, I) \right).$$

We then perform a softmax selection of M infosets based on the gains $[\Gamma_I]_{I \in \hat{\mathcal{I}}_j}$. BR policy π^*_j is then added as an action edge to each of these infosets. The process creates new infosets, depending on the observable effects of the abstract policy. We illustrate how the empirical game tree is extended by this method in App. B, using Bargain as an illustrative example.

Finally, TE-PSRO updates payoff estimates for the augmented $\hat{G}$ by simulating the strategy combinations that result from the newly added edges and recording the sampled payoffs. All epochs are allocated the same total number of gameplay simulations, called the *simulation budget*, distributed equally among all *new* strategy profiles. Thus, the number of samples per profile is fixed based on the TE-PSRO epoch, independent of the choice of M.

5 COMPUTING REFINED NASH EQUILIBRIA

Tree-based game models afford consideration of solution concepts specific to the extensive form. We specifically investigate the use of ***subgame perfect equilibrium*** (SPE), a refinement of NE that rules out solutions containing non-credible threats. To make use of SPE, we need a definition that applies to games of imperfect information, and an algorithm that computes such solutions.

Definition 5.1. A ***subgame*** of game G is a directed rooted subtree given by $G' = \langle N, H', V', \{\mathcal{I}'_j\}^n_{j=0}, \{A'_j\}^n_{j=1}, X', P', u' \rangle$ satisfying the following:

- The root h' of tree H' must be the only node in its information set.
- As a subtree of H, H' must include all nodes in H that succeed h'.
- For any $j \in N$ and for all $I \in \mathcal{I}_j$, if $I \in \mathcal{I}'_j$, then the nodes $h \in I$ must all be part of H'; if $I \notin \mathcal{I}'_j$, then all its nodes must be part of $H \setminus H'$.
- V', $\{\mathcal{I}'_j\}^n_{j=0}$, $\{A'_j\}^n_{j=1}$, X', P', and u' are restrictions to H' of V, $\{\mathcal{I}_j\}^n_{j=0}$, $\{A_j\}^n_{j=1}$, X, P, and u, respectively.

Definition 5.2 ([14]). A ***subgame perfect equilibrium*** (SPE) of game G is an NE of G that also induces NE play in each of G's subgames.

In finite perfect-information EFGs, an SPE always exists in pure strategies and can be readily computed using the classic backward induction approach. With imperfect information, however, a subgame may not admit a pure-strategy NE at all. Kaminski [7] proposed the ***generalized backward induction*** (GBI) approach for finding the set of SPE for a potentially infinite game of imperfect information. A key feature of GBI is the re-expression of the game tree as a set of its proper subgames organized by their roots. Other crucial implementation details are not fully specified in the original article, in particular how to identify NE of subgames that include non-singleton information sets; a naïve implementation using exhaustive enumeration of strategy profiles for combinations of subgames has a runtime that is exponential in game size.

We provide a practical, modular algorithm for finding an SPE via GBI in a finite, imperfect-information EFG. Our algorithm combines dynamic programming with a Nash solver subroutine, using Kaminski's [2019] idea of organizing the game into subgames. Alg. 1 presents our method. ComputeSPE uses subroutines described here at a high level (see App. C for full pseudocode).

Algorithm 1 : ComputeSPE

Require: Input game G
1: $\Psi \leftarrow$ GetSubgameRoots(G)
2: $\ell \leftarrow$ height of h_0 in Ψ
3: $\{\Theta_k\}^\ell_{k=1} \leftarrow$ GetSubgameGroups(G, Ψ, ℓ)
4: $\boldsymbol{\sigma}^{SPE} \leftarrow$ GetInitialSPE(G, Θ_1)
5: **for** $1 < k \leq \ell$ **do**
6: **for** $\theta \in \Theta_k$ **do**
7: Extract $\boldsymbol{\sigma}^{SPE}|_{G_\theta} \leftarrow \{\boldsymbol{\sigma}^{SPE}(I) \mid I \in G_\theta \cap \boldsymbol{\sigma}^{SPE}\}$
8: $\boldsymbol{\sigma}^{SPE} \leftarrow \boldsymbol{\sigma}^{SPE} \cup$ NashSolver(G_θ, $\boldsymbol{\sigma}^{SPE}|_{G_\theta}$)
9: **end for**
10: **end for**
 return $\boldsymbol{\sigma}^{SPE}$

We first call GetSubgameRoots to find the roots of all subgames in the input game G and arrange them into a tree Ψ rooted at h_0, the root of G. A root in Ψ has a ***height*** $1 \leq k \leq \ell$ where the subgames closest to the leaves of G have height 1 and h_0 has height ℓ. To find the roots, it is sufficient to check which nodes in H are roots of subtrees satisfying the conditions of Definition 5.1. GetSubgameGroups collects all subgame roots in Ψ at height k into a set Θ_k, for $1 \leq k \leq \ell$. Then, we use dynamic programming to iterate over the subgames of each Θ_k and solve each subgame at height k directly via a chosen NashSolver. The union $\boldsymbol{\sigma}^{SPE}$ of all SPE found for the subgames in G with height less than k is updated with each new partial SPE. G_θ denotes the subgame rooted at node $\theta \in \Theta_k$. The union of all SPE across all subgames in G by definition must be the SPE of G. In order to avoid overwriting the SPE that have been computed for any subgames at smaller heights in G_θ, we pass the partial SPE $\boldsymbol{\sigma}^{SPE}|_{G_\theta}$ in as input to NashSolver and restrict the solver to find a solution only for the information sets within G_θ that are not already included in $\boldsymbol{\sigma}^{SPE}|_{G_\theta}$. This ensures that the runtime of ComputeSPE is linear with respect to the

size of G and thus scalable modulo the runtime of NashSolver (see App. D for runtime analysis). For our experiments (§6), we devised an adaptation of the counterfactual regret (CFR) minimization algorithm [20], called SubgameCFR, as our NashSolver.

6 EXPERIMENTS

We now report experiments that we conducted to evaluate our TE-PSRO approach by applying it to the two games described in §3.

6.1 Parameter settings

For Bargain (§3.1), we set $\tau = 3$, $\bar{V} = 10$, $\nu = 5$, $\gamma = 0.99$, $T = 5$, and $n \in \{5, 6, 7\}$. We generated five unique sets of the remaining parameters $\mathbf{p}$, $(\mathbf{v}_1, \mathbf{v}_2)$, P_1, P_2 uniformly at random from their respective supports in order to evaluate TE-PSRO's performance on a variety of game instances. For GenGoof (§3.2), we set $K = 4$, $u_{\max} = 10$, calling this instance GenGoof$_4$. The simulator budget was 100 samples for Bargain and 200 samples for GenGoof$_4$.

We ran all experiments on our local computing cluster using a single core. Runtime and memory requirements depend on the choice of $M \in \{1, 2, 4, 8, 16\}$, which determines the rate of growth of $\hat{G}$ across TE-PSRO epochs (see App. E for details). Unless otherwise stated, every experiment was performed for five randomly seeded trials for each setting. Error bars in our plots correspond to a 95% confidence interval.

6.2 Results

Our first set of experiments assesses space requirements for the empirical game $\hat{G}$ as a function of the number M of infosets augmented per epoch. At each epoch of TE-PSRO, we recorded the total number of information sets across players in $\hat{G}$ and the memory required by the emprical model. We report the average empirical game size for each of the two games studied in terms of the number of player information sets of all players and memory required in megabytes (MB) in Fig. 3 and Fig. 15 in App. F.1 respectively, for representative values of M; each curve for Bargain is averaged over 100 trials per value of M across all five sets of bargaining parameters. The broad takeaway from all plots in this set is the following. Although the rate of increase in the size of the $\hat{G}$ steepens with M in the plots, its size is still manageable after many epochs of TE-PSRO. If we had added a new policy to all information sets rather than to only a subset of size M, $\hat{G}$ would grow to as many as 3000 or 4000 information sets after only five epochs of TE-PSRO, which is an unsustainable trajectory. See App. F.1 for further insights on the difference between the two games.

Our second set of experiments provides evidence for the effectiveness of our algorithm ComputeSPE(§5) as well as the non-triviality of obtaining an SPE of an imperfect-information extensive-form game. We ran two suites of TE-PSRO on each of Bargain and GenGoof$_4$: each suite consisted of 50 trials for each M setting, using NE as the MSS for 25 trials and SPE as the MSS for the remaining 25. In one suite, we evaluated intermediate and final $\hat{G}$ (EVAL in Fig. 1) by NE computed using CFR, and the other by SPE using Alg. 1. We computed the regret of the respective solutions with respect to each subgame of $\hat{G}$ and reported the maximum over subgames, that is, the ***worst-case subgame regret***; a solution with a lower value of this quantity is a better SPE approximation. Thus, Figs. 4b and 4d

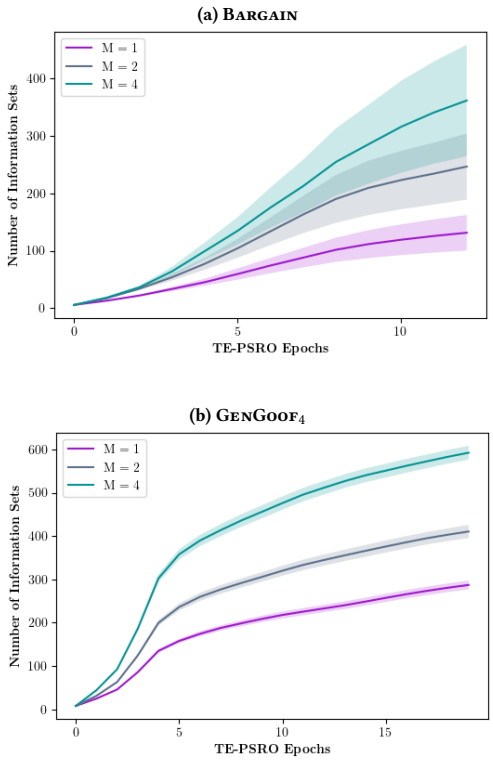

Figure 3: Total number of information sets of both players in empirical game $\hat{G}$ over the course of TE-PSRO's runtime, averaged over all combinations of all parameters (except M) and seeds.

verify that Alg. 1 does indeed produce an approximate SPE Figs. 4a and 4c demonstrate that our regular Nash solver does not happen to stumble upon NE that are also subgame-perfect. Additionally, the worst-case subgame regret increases with the complexity of $\hat{G}$, reflected in both setting of M and epochs of TE-PSRO. Note that these experiments are not for gauging the quality of the models produced by TE-PSRO; instead, TE-PSRO is used to generate a sequence of empirical games of increasing size and complexity (in terms of the number of non-singleton information sets) that serve as more and more challenging test cases for our game-solving algorithms.

Our final experiment set characterizes TE-PSRO performance in terms of the MSS choice (SPE vs. NE) and different values of M. To compare MSS choices, we computed the profile regret (§2) with respect to the underlying game of the solution $\boldsymbol{\sigma}^*$ returned by EVAL in each epoch of TE-PSRO epoch; we used both NE and SPE as EVAL, giving us two sets of comparison metrics.

Fig. 5 depicts our average regret results for Bargain under various settings. We ran TE-PSRO for 25 trials per value of M and four combinations of choices for EVAL and MSS. In each trial, TE-PSRO was allowed to run for at most 30 epochs, terminating early when the computed best responses did not yield an improvement greater than 0.1 over the current solution $\boldsymbol{\sigma}^*$. Fig. 5a shows that TE-PSRO outperforms the normal-form version NF-PSRO, regardless of EVAL/MSS choice, even for $M = 1$. Figs. 5b and 5c show that

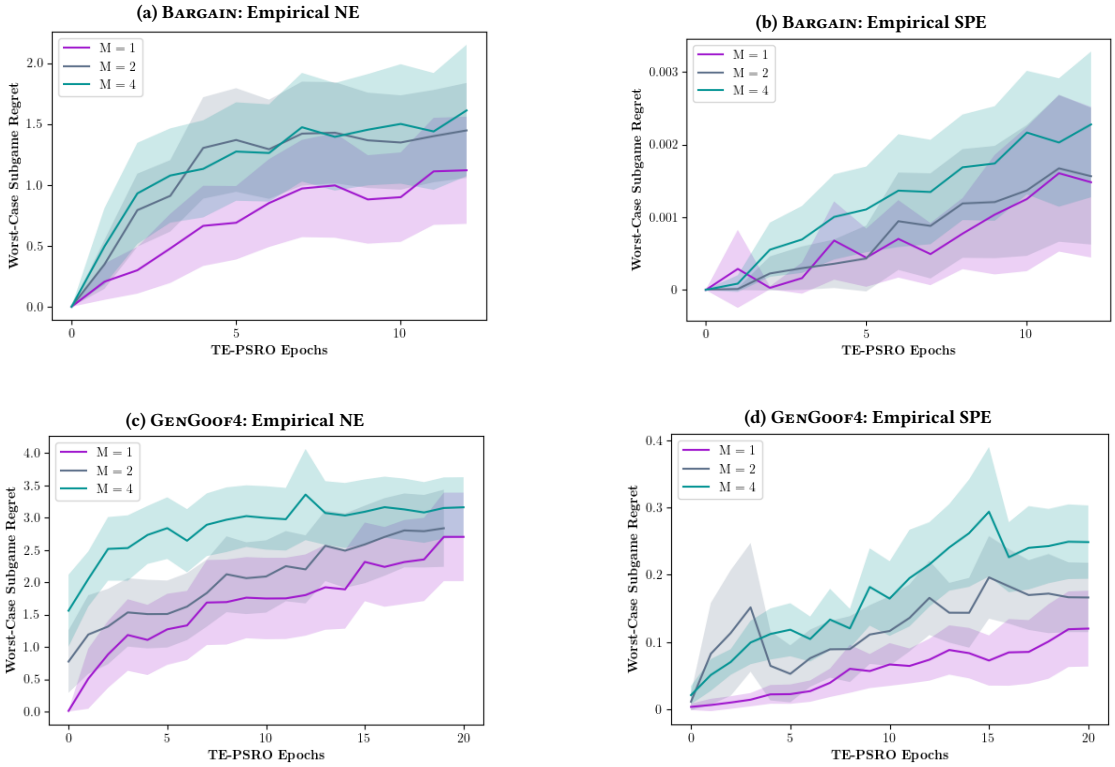

Figure 4: Average worst-case subgame regret of NE and SPE solutions to empirical game $\hat{G}$ for the two games studied. Note that the scale of vertical axis of (b) is finer than that of (a) by a factor $\approx 10^3$ while the corresponding factor ≈ 10 for (d) and (c).

SPE beats NE as an MSS for $M \in \{4, 8\}$, converging faster to near-zero regret regardless of EVAL. Finally, Fig. 5d compares results for different M settings, with NE as EVAL and SPE as MSS. The main takeaway is that intermediate values $M = 4$ and $M = 8$ outperform lower and higher settings. Intuitively, a higher M produces $\hat{G}$ with broader coverage earlier, but with a fixed sampling budget, each tree path is estimated less accurately, resulting in a non-monotonic performance with respect to M. App. F presents the full set of plots over combinations of EVAL, MSS, and M.

We assessed the statistical significance of the MSS comparisons for Bargain using a permutation test, for each setting of M and EVAL. Our figure of merit is the area $\mathcal{A}_{\text{MSS}}$ under the regret curve, calculated starting from TE-PSRO epoch 5 to avoid the noisy startup phase. Our test statistic is $\Delta_{\text{MSS}} = \mathcal{A}_{\text{NE}} - \mathcal{A}_{\text{SPE}}$, resampled over 1000 permutations of the MSSs. The p-value is the fraction of times the difference under permutation (i.e., a null hypothesis that the MSSs are equally effective) is greater than the observed Δ_{MSS}. For $M = 4$, the superiority of SPE as MSS is significant ($p = 0.007$) with NE as EVAL. For $M = 8$, SPE as MSS is significantly better both for NE ($p = 0.006$) and SPE ($p = 0.021$) as EVAL. The results are less consistent and less significant for non-optimal M values (App. F.5).

For experiments on GenGoof$_4$, we additionally constrained the space of empirical game models induced by TE-PSRO by coarsening away the stochastic events in the last one two rounds of the full three-round underlying game. We indicate this by IR that stands for ***included rounds***; it may take values $[0]$, $[0, 1]$ or $[0, 1, 2]$ indicating that each empirical game tree can include (a) stochastic event(s) only in its first round, first two rounds, and all three rounds respectively. In Fig. 6, we see that $IR = [0]$ and $IR = [0, 1]$ tended to yield the best performance regardless of MSS and that, for both of these settings, SPE outperformed NE as the MSS. App. F.6 offers insights on how IR and M jointly impact TE-PSRO performance.

7 CONCLUSIONS

We introduced multiple extensions of Tree-Exploiting PSRO, enabling its application to complex games of imperfect information. Our main innovation is the treatment of best responses computed by DRL as abstract policies, incorporated as actions in the empirical game tree. To manage growth of the empirical game as BRs are generated over the course of TE-PSRO, we introduced a hyperparameter M which controls the number of infosets that can be expanded per epoch. Finally, we demonstrated that having an extensive-form empirical game model can be leveraged in the form of new meta-strategy solvers based on Nash refinements. Toward that end, we developed a modular algorithm for identifying SPE solutions in imperfect-information games. We demonstrated these methods on two carefully constructed complex games, featuring multiple rounds of offer/counteroffer with signaling options. We

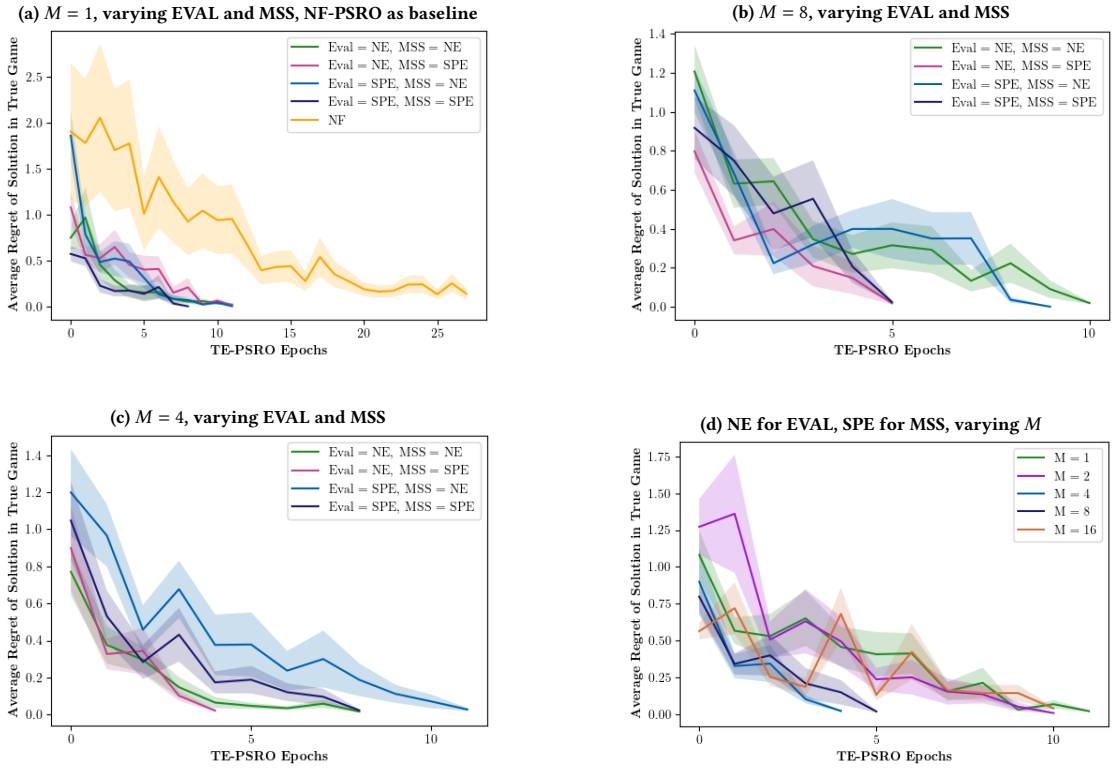

Figure 5: Average regret of solution σ^* of empirical game for Bargain over iterations of TE-PSRO, using NE or SPE as the MSS or EVAL and different values of M.

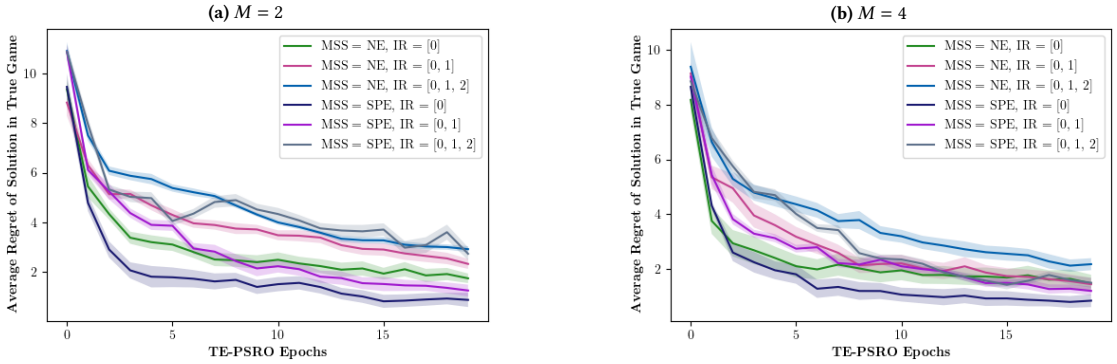

Figure 6: Average regret of σ^* evaluated in GenGoof$_4$ over the course of TE-PSRO's runtime, using NE or SPE as the MSS.

showed that TE-PSRO easily outperforms normal-form PSRO in this environment, and that intermediate values of M perform best.

Particularly intriguing is our finding that exploiting Nash refinement in an MSS offers promise for improving strategy exploration. Even when the goal is not to find a subgame-perfect solution (i.e., EVAL is NE rather than SPE), targeting best response to SPE rather than NE can be beneficial. Further work is required to confirm the scope of and understand the reasons for this advantage. One intuitive explanation is that empirical game equilibria containing non-credible threats may be particularly exploitable in the underlying game and thus not the most promising lines to pursue in strategy exploration.

ACKNOWLEDGMENTS

This work was supported in part by a grant from the Effective Altruism Foundation and by the US National Science Foundation under CRII Award 2153184.

REFERENCES

[1] Ariyan Bighashdel, Yongzhao Wang, Stephen McAleer, Rahul Savani, and Frans A. Oliehoek. 2024. Policy space response oracles: A survey. In *33rd Int'l Joint Conference on Artificial Intelligence* (Jeju, Korea). 7951–7961.

[2] Branislav Bošanský, Viliam Lisý, Marc Lanctot, Jiří Čermák, and Mark H.M. Winands. 2016. Algorithms for computing strategies in two-player simultaneous move games. *Artificial Intelligence* 237 (2016), 1–40.

[3] David DeVault, Johnathan Mell, and Jonathan Gratch. 2015. Toward natural turn-taking in a virtual human negotiation agent. In *AAAI Spring Symposium on Turn-Taking and Coordination in Human-Machine Interaction.*

[4] Shaheen Fatima, Sarit Kraus, and Michael Wooldridge. 2014. *Principles of Automated Negotiation.* Cambridge University Press.

[5] James F. Fixx. 1972. *Games for the Superintelligent.* Doubleday.

[6] Johannes Heinrich and David Silver. 2016. Deep reinforcement learning from self-play in imperfect-information games. arXiv:1603.01121 [cs.LG]

[7] Marek Mikolaj Kaminski. 2019. Generalized backward induction: Justification for a folk algorithm. *Games* 10 (2019). Issue 34.

[8] Christine Konicki, Mithun Chakraborty, and Michael P. Wellman. 2022. Exploiting extensive-form structure in empirical game-theoretic analysis. In *18th Int'l Conference on Web and Internet Economics.* 132–149.

[9] Marc Lanctot, Vinicius Zambaldi, Audrunas Gruslys, Angeliki Lazaridou, Karl Tuyls, Julien Pérolat, David Silver, and Thore Graepel. 2017. A unified game-theoretic approach to multiagent reinforcement learning. In *31st Annual Conference on Neural Information Processing Systems.*

[10] Mike Lewis, Denis Yarats, Yann N. Dauphin, Devi Parikh, and Dhruv Batra. 2017. Deal or no deal? End-to-end learning for negotiation dialogues. In *2017 Conference on Empirical Methods in Natural Language Processing.* 2443–2453.

[11] Zun Li, Marc Lanctot, Kevin R. McKee, Luke Marris, Ian Gemp, Daniel Hennes, Paul Muller, Kate Larson, Yoram Bachrach, and Michael P. Wellman. 2023. *Combining tree-search, generative models, and Nash bargaining concepts in game-theoretic reinforcement learning.* Technical Report 2302.0079. arXiv.

[12] Stephen McAleer, John Lanier, Kevin A. Wang, Pierre Baldi, and Roy Fox. 2021. XDO: A double oracle algorithm for extensive-form games. In *35th Annual Conference on Neural Information Processing Systems.*

[13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529–533.

[14] R. Selten. 1975. Reexamination of the perfectness concept for equilibrium points in extensive games. *International Journal of Game Theory* 4, 1 (1975), 25–55.

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

[16] Karl Tuyls, Julien Perolat, Marc Lanctot, Edward Hughes, Richard Everett, Joel Z. Leibo, Csaba Szepesvári, and Thore Graepel. 2020. Bounds and dynamics for empirical game-theoretic analysis. *Autonomous Agents and Multi-Agent Systems* 34, 7 (2020).

[17] Yongzhao Wang, Qiurui Ma, and Michael P. Wellman. 2022. Evaluating strategy exploration in empirical game-theoretic analysis. In *21st Int'l Conference on Autonomous Agents and Multi-Agent Systems.* 1346–1354.

[18] Michael P. Wellman. 2016. Putting the agent in agent-based modeling. *Autonomous Agents and Multi-Agent Systems* 30 (2016), 1175–1189.

[19] Michael P. Wellman, Karl Tuyls, and Amy Greenwald. 2024. Empirical Game-Theoretic Analysis: A Survey. *Journal of Artificial Intelligence Research* (2024).

[20] Martin Zinkevich, Michael Johanson, Michael H. Bowling, and Carmelo Piccione. 2007. Regret minimization in games with incomplete information. In *21st Conference on Neural Information Processing Systems.*