Public Information representation for Adversarial Team Games

Luca Carminati, Federico Cacciamani, Nicola Gatti
Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano
Piazza Leonardo da Vinci, 32, 20133, Milano, Italy
luca5.carminati@mail.polimi.it, {federico.cacciamani, nicola.gatti}@polimi.it

Abstract

In this work, we focus on a team of agents playing a sequential game against a single adversary. The presence of asymmetric information among the members of the team makes the problem of computing a solution hard even with zero-sum payoffs. A number of *ad hoc* algorithms available in the literature tackle this problem resorting to *Linear Programming*. Our novel approach consists in using a procedure to convert the game to a classical *two-player zero-sum* game. In this converted game, the team is transformed into a single coordinator player which only knows information common to the whole team and prescribes to the players an action for any possible private state. We named this procedure *Public Team Conversion*, and its result is an extensive-form game maintaining most of the structure of the original game. Our conversion allows to adopt the highly scalable and performant techniques already developed for two-players zero sum games, including techniques for generating automated abstractions. Because of the NP-hard nature of the problem, the Public Team Conversion produces a game which may be exponentially larger than the original one. We then provide three pruning techniques to reduce the exponential increase of size to its square root. Finally, we present experimental results obtained by applying our technique to standard benchmarks in the field, *Kuhn Poker* and *Leduc Poker*. We also apply state of the art equilibrium computation algorithms on the resulting game, showing the effectiveness of our approach.

Introduction

Research efforts on imperfect information games traditionally focused on two player zero sum (2p0s) scenarios, in which two agents act on the same environment, receiving opposite payoffs. In this setting, superhuman performances have been achieved even in large instances of games, such as *Poker Hold'em* by (Moravcík et al. 2017), (Brown and Sandholm 2017), (Brown and Sandholm 2019b), and *Starcraft II* by (Vinyals et al. 2019).

On the other hand, the design of techniques to produce robust agents also when multiple agents act in the same environment, possibly sharing their rewards, is still an open challenge.

Copyright © 2021, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

In this work, we focus on adversarial team games in which a team of two agents cooperates against a common adversary. Specifically, we focus on the *ex ante coordination* scenario: in this setting the team members agree on a common strategy beforehand and commit to it during the game, without communicating any further. Examples of such a scenario are collusion in poker games, the defenders in the card-playing stage of *Bridge*, and a team of drones acting against a single adversary. In these scenarios, a team of agents shares the same payoffs and coordinates against a single adversary having an opposite payoff, in face of private information separately given to each team member.

A natural way to characterize the desired behavior by each agent is the *team-maxmin* equilibrium (TME) solution concept, defined by Von Stengel in (Von Stengel and Koller 1997). In (Celli and Gatti 2018), this concept is extended to extensive-form games considering different means of communication. The team-maxmin equilibrium with correlation (TMECor) characterizes optimal rational behavior in the *ex ante* coordination scenario we consider in this paper. While the existance and uniqueness of value of a TMECor is guaranteed, the computation of an equilibrium corresponding to this solution concept is proven to be an inapproximable, NP-hard problem (Celli and Gatti 2018). Therefore, the main open challenge is to efficiently find a strategy profile corresponding to a TMECor in a generic adversarial team game.

Related works. Classical algorithms already developed for 2p0s games cannot be applied directly to team games since the private information held by each team member cannot be shared with the other team members; therefore the problem of finding a TMECor in an adversarial team game has been traditionally resolved through the use of *ad hoc* techniques. Two main directions have been followed by previous approaches: mathematical programming and reinforcement learning techniques.

Mathematical programming approaches are generally based on a *column generation* approach. They constrain the team to play a probability distribution over a finite set of correlated plans, and iteratively add new correlated plans giving the maximum increase in value for the team until no new plans can be added. The constraint to draw plans from a finite set for the team allows a polynomial-time formulation of the TMECor problem, while the plan to add is

determined by using an oracle working in an implicit formulation of the normal form representation of the original game, in which the private information of a team player can be ignored during the selection of a plan. *Hybrid Column Generation* (HCG) (Celli and Gatti 2018) itereatively solves two linear programs to determine the TMECor stategy profile of the team and the adversary considering the limited set of plans available, while uses a integer linear program to find the best response joint plan to add to the currently considered plans given the current strategy of the adversary. *Fictitious Team Play* (FTP) (Farina et al. 2018) is instead a *fictitious play* (Brown 1951) procedure running on a modified representation of the original game; in this representation one of the two team members selects a pure strategy at the start of the game, and the other team member plays against the adversary in the original game considering his teammate fixed as specified by their chosen plan. *Faster Column Generation* (FCG) (Farina et al. 2021) is a column generation algorithm similar to HCG working in a more efficient semi-randomized correlated plans representation, prescribing a pure strategy for one team member and a mixed strategy for the other. This representation also allows a cost-minimizing formulation of the best response problem as a dual of the program to find a TMECor considering the restricted set of plans. Overall, those procedure are general and simple to implement, but the use of linear programs strongly limits the scalability of these exact formulations.

Reinforcement learning approaches avoid considering plans over the entire game, and instead explicitly model the correlating coordination signal in the original extensive form adversarial team game. *Soft Team Actor-Critic* (STAC) (Celli et al. 2019) fixes a number of possible uniform signals at the start of the game and uses a modified actor-critic procedure to converge to a strategy for each player for each signal. *Signal Mediated Strategies* (SIMS) (Cacciamani et al. 2021) works in a *perfect recall refinement* of the original game, populates a buffer of trajectories sampled from the optimal strategy for the joint team against the adversary in this refined game, and learns a distributed strategy for each team member in the original game from the trajectories in this buffer. Overall, reinforcement learning techniques are more scalable than the mathematical programming ones, but present convergence issues.

Original contributions. We propose an algorithmic procedure, called PUBLICTEAMCONVERSION, to convert a generic instance of a team game into a 2p0s game, and provide three pruning techniques to reduce the size of the converted game. We formally prove that a Nash Equilibrium in the converted game corresponds to a TMECor in the original game. In addition, we present vEFG, an alternative description of extensive-form games better suited for the characterization of public information.

The core idea of our approach is to combine the sharing of all team members' strategies with the use of public information to determine a belief over the possible private information of all team members. As the game progresses, the information state of each player is enriched by the notion that other team members may or may not be in a specific private state given the deterministic strategy shared at the start of the game and the observed public actions played. In fact, playing a specific action when the overall strategy is known, communicates part of the private information. Following this intuition, we propose an explicit representation for the game by joining the team members into a single coordinator who has an information state based only on the public information shared among team members and that prescribes a deterministic action for each possible private state of the player.

Our work builds upon Nayyar et al. in (Nayyar, Mahajan, and Teneketzis 2013); similarly to our work, a shared coordinator, living in the public information state of the team, prescribes an action to the team members for any possible private state allowing the training of a common strategy for the team. This coordinator-based transformation has been employed in recent works in the fully cooperative setting of *Hanabi* (Foerster et al. 2019; Sokota et al. 2021). However, the original transformation proposed focuses on the context of decentralized stochastic control in a fully cooperative scenario, whereas we generalize such coordinator transformation in an adversarial team game setting.

Preliminaries

In this section, we provide a formal introduction to the notation used in the paper, mainly derived from the one presented in (Kovařík et al. 2020), and define the crucial concepts of Nash Equilibrium and TMECor.

An *extensive form game with imperfect information* (EFG) $\mathcal{G}$ is a tree model representing a sequential finite interaction among different players, obtaining a payoff at the end of each trajectory of play. The *set of players* is $\mathcal{N} = \{1, \dots, N\}$, and we use c to indicate the *chance player*. The chance player represents Nature in the game, and plays a fixed strategy σ_c. Given a player p, we use $-p$ to indicate all the players different from p, chance included. The *action set* is indicated as $\mathcal{A} = \bigcup_{p \in \mathcal{N}} \mathcal{A}_p$, and each node h of the tree is identified by the tuple of actions played to reach this point of the game, called *history*. Similarly, the *set of histories* is denoted by $\mathcal{H} = \bigcup_{p \in \mathcal{N}} \mathcal{H}_p$. Given two histories h, g we indicate with $h \sqsubset g$ if h is a parent history of g. $\mathcal{Z} = \{h \in \mathcal{H} : \nexists g, h \sqsubseteq g\}$ is the set of *terminal histories*, also called leaf histories, and $u_p(z) : \mathcal{Z} \to \mathbb{R}$ is the utility of player p upon reaching terminal history z. $\mathcal{N}(h)$ and $\mathcal{A}(h)$ indicate respectively the player playing and the actions available to play in node h. ha indicates the history reached by playing action a in h. To indicate the information state of a player p, we define $\mathcal{I}_p$ as a partition over $\mathcal{H}_p$, indicating the nodes which are indistinguishable for a player. $I \in \mathcal{I}_p$ is called *information set* and we have the constraint that $\forall h, h' \in I : \mathcal{A}(h) = \mathcal{A}(h')$. We use $\mathcal{A}(I)$ to indicate the actions available in a information state and $\mathcal{I}(h)$ to indicate the information set of a history.

In this setting, we have different options to represent a strategy of each player. A *behavioral strategy* $\sigma_p(I) : \mathcal{I}_p \to \Delta^{|A(I)|}$ is a function associating to each information set of player p a probability distribution over the possible actions. A *normal form plan* (also called plan or pure strategy) $\pi_p \in \Pi_p := \times_{I \in \mathcal{I}_p} \mathcal{A}(I)$ for player p is a tuple specifying an

action for each information set of p. A *normal form strategy* $\mu_p \in \Delta^{|\Pi_p|}$ is a probability distribution over plans.

An EFG has *perfect recall* if for every $p \in \mathcal{N}$, for every $I \in \mathcal{I}_i$ and any pair $h, h' \in I$, the sequences of infosets and actions of player p leading to h, h' are the same.

For a EFG, a deterministic timing is a labelling of the nodes in $\mathcal{H}$ with non-negative real numbers such that the label of any node is at least one higher than the label of its parent. A deterministic timing is exact if any two nodes in the same information set have the same label. A EFG is *timeable* if it admits a deterministic exact timing. A EFG is *1-timeable* if it admits a deterministic exact timing such that each node's label is exactly one higher than its parent's label.

A *strategy profile* is a tuple associating a strategy with each player. The expected value for a player can be defined as $u_p(\boldsymbol{\sigma}) = u_p(\sigma_p, \boldsymbol{\sigma_{-p}}) = \sum_{z \in \mathcal{Z}} \rho^{\boldsymbol{\sigma}}(z) u_p(z)$, where $\rho^{\boldsymbol{\sigma}}(z)$ indicates the probability of reaching terminal node z given the strategies of all players. Any two strategies of a player are called *payoff equivalent* if, for any strategy of the other players, both strategies have the same expected payoff. Given a player p and a strategy for all the other players $\boldsymbol{\sigma_{-p}}$, a *best response strategy* $BR(\boldsymbol{\sigma_{-p}})$ for p is a strategy maximizing their expected payoff. A *Nash Equilibrium* (NE) is a strategy profile $\boldsymbol{\sigma}$ such that no player can gain a larger payoff value by deviating to a different strategy. Formally, $\forall p \in \mathcal{N} : \sigma_p \in BR(\boldsymbol{\sigma_{-p}})$ A *Team Maxmin equilibrium with correlation* (TMECor) is a TME equilibrium in which all team players are correlated. Formally, a TMECor is a strategy profile $\boldsymbol{\sigma}^*$ such that: $\boldsymbol{\sigma}^* = \arg\max_{\boldsymbol{\sigma_T} \in \Delta^{|\times_{p \in \mathcal{T}} \Sigma_p|}} \min_{\sigma_o} u_{\mathcal{T}}(\boldsymbol{\sigma_T}, \sigma_o)$

vEFG Representation

Characterizing the public information for a set of players is not possible with the standard EFG notation, since the information structure is solely defined by its information set partitions, and it is difficult to determine common patterns in the partitions of multiple players. In this section, we therefore introduce a refined formalization derived from EFGs called *extensive form games with visibility* (vEFG). A vEFG is identical to an EFG, apart from the introduction of an explicit visibility function $Pub_p(a)$, describing whether an action is detected by player p or not. The information state of a player in a history is determined by considering the actions of that history that are visible to him. This allows to make the information known to a player more explicit than in traditional EFGs.

The *visibility function* is formally defined as $Pub_p(a) : \mathcal{A} \to \{\text{seen}, \text{unseen}\}$.

A vEFG has *perfect recall* if any action played by a player is seen by the player himself.

The visibility function can also be extended to sets of players. $Pub_A(a) : \mathcal{A} \to \{\text{pub}, \text{priv}, \text{hidden}\}$, such that:

$Pub_A(a) = \text{pub} \iff \forall p \in A : Pub(a) = \text{seen}$

$Pub_A(a) = \text{hidden} \iff \forall p \in A : Pub_p(a) = \text{unseen}$

$Pub_A(a) = \text{priv}$ otherwise

From the visibility function it is then possible to derive the classical information set structures and the notion of public tree.

The information set structure $\mathcal{I} = (\mathcal{I}_p)_{p \in \mathcal{N}}$ can be recovered from Pub_p, by considering in the same infoset the histories corresponding to the same sequence of seen actions from a player. Formally, $\forall h, h' \in \mathcal{H} : h, h' \in I \subset \mathcal{I}_p$ if and only if $\mathcal{N}(h) = \mathcal{N}(h') = p$ and $(a)_{a \in h : Pub_p(a) = \text{seen}} = (a')_{a' \in h' : Pub_p(a') = \text{seen}}$

$\mathcal{S}$ is the *public tree* associated to the game. $S \in \mathcal{S}$ is called *public state*. Two histories belong to the same public state if they share the same public actions (considering every player) and differ only by the private actions. Formally, h, h' belong to the same public state iff $(a)_{a \in h : Pub_{\mathcal{N}}(a) = \text{pub}} = (a')_{a' \in h' : Pub_{\mathcal{N}}(a') = \text{pub}}$. We also use $\mathcal{S}(h)$ to indicate the public state associated to a history. Moreover, we can also define the public tree for a subset of players. Given a subset of players $X \subset \mathcal{N}$, then $\mathcal{S}_X$ is the public tree associated to players in X.

We conjecture that the set of vEFGs with perfect recall is equivalent to the set of EFGs with perfect recall. Note that in vEFG we have no notion of forgetting actions, thus imperfect recall situations in which an observation is forgotten by a player cannot be represented.

In order to guarantee a specific structure of the vEFGs we treat, we introduce the concept of *public turn-taking*, characterizing games in which the sequence of players acting is common knowledge across all players. Intuitively, each player knows when other players played an action in the past, even if the game has imperfect information and the specific action played may be hidden. This is a refinement to the concept of 1-timeability, in which not only the length of the history is identical but also the sequence of players.

Definition 1 (Public turn-taking property)**.** *A vEFG is public turn-taking if:*

$$\forall I \in \mathcal{I}, \forall h, h' \in I : (\mathcal{P}(g))_{g \sqsubseteq h} = (\mathcal{P}(g'))_{g' \sqsubseteq h'}$$

Lemma 1 (Transformation into public turn-taking game)**.** *Any vEFG can be made public turn-taking by adding player nodes with a single noop action. This modification will not increase the size of the game by a factor larger than* $(|\mathcal{N}| + 1)|\mathcal{H}|^2$.

Information Structure in Team Games

The core problem of finding a TMECor in adversarial team games resides in *asymmetric visibility*, since team members have a private state that does not allow to create a joint coordination player.

In the following we characterize the possible types of asymmetric visibility that may cause imperfect recall for the joint player, and singularly address them.

- **Non-visibility over a team member's action**. If a team member plays a action which is hidden from another team member, the joint team player would have imperfect recall due to forgetting his own played actions. This source of imperfect recallness can be avoided in a TMECor by considering the shared deterministic strategies before the game starts. This allows to know a priori which are the exact actions played by team members in each node. Thus it is safe to apply a perfect recall refinement in the original

game, which corresponds to always consider the chosen action of a team member as seen by other team members.

- **Non-visible game structure**. Consider two nodes in the same information set for a player before which the other team member may have played a variable number of times, due to a chance outcome non visible to the team member of this nodes. In this case a perfect recall refinement is not applicable to distinguish the nodes, because it would give the joint coordinator a visibility not correspondent to the one of the players in the game. To solve this edge case, we require the property of public turn-taking.
- **Private information disclosed by chance/adversary to specific team members.** It is the most complex type of non visibility, since in a TMECor we have no explicit communication channels through which share information, and therefore this type of joint imperfect recall can only be addressed by considering a strategically equivalent representation of the game in which at most one of the team players has private information.

State of the art techniques address this issue by considering a richer representation of the original game, to make more explicit the information needed for an effective team coordination.

Linear programming (LP) approaches use a implicit normal form representation for one or both the team members. This because a normal form plan allows one to specify the complete strategy of a player in one shot, and by sharing this information with the other team member (or equivalently, by correlating each player's plan selection), they can optimize their strategy knowing how the other member will behave for the rest of the game. This solves the coordination problem, at the cost of large action space for the normalized players, which however is never explicitly represented thanks to the column generation procedure.

Reinforcement learning (RL) approaches work on a refined representation of the original game, without any normalization. This avoids the exponential growth of the size of the game representation. However, such game representation lacks enough information to fully characterize the coordination among team members. In the case of STAC, the fragmented and fixed system of signals severely affects the convergence performances, while SIMS guarantees convergence only on instances without private observations, which is not the case of most practical team coordination problems.

The main theoretical advance of the present work is the use of a transformation based on public information of the team to solve this problem of private information.

Our PUBLICTEAMCONVERSION transformation builds upon the one proposed by (Nayyar, Mahajan, and Teneketzis 2013). The main idea is to substitute team players in the original game with a coordinator that only sees public information among team members. The coordinator prescribes the action of the currently playing team member by emitting a prescription directed to them, associating each possible private state of the player with one possible action. In this way, each player becomes a *puppet agent* which executes the prescriptions received by using the private information at his disposal. The important feature is that such a coordinator only lives in the space of public information of the team, and therefore the decision points giving prescriptions to each team player can be associated to a unique player.

Intuitively, we can see this process of prescription as an explicit weak normalization. In fact, we maintain the public tree structure of the original game, while we normalize each public state, in the sense that we build smaller local plans with an action for each private state corresponding to a public state. As in the normal form conversion, we still have an exponential explosion due to the combination of actions for each private state, but the decomposition per public state makes it more computationally tractable, especially when employing pruning techniques.

Public Team Conversion Algorithm

We now present the proposed algorithmic procedure to convert an adversarial team game into a 2p0s game, in which a coordinator player takes the strategic decision on behalf of the team.

The algorithm recursively traverses the original game tree in a post-order depth-first fashion. For each traversed node, corresponding nodes are instantiated in the converted game. The conversion procedure works by copying unaltered all the chance, terminal, and adversary nodes, by considering that the visibility of the new coordinator player t is the public visibility among team members.

Team member nodes are instead mapped to a new coordinator node, in which the coordinator plays a prescription Γ among the combination of possible actions for each information state I belonging to the public team state of the node to convert. The effect of playing a prescription in the converted game corresponds to having played the action prescribed for the current state in the original game. A prescription is only visible to the team members. To have the players see the action played in the original game, we add a dummy chance node playing that action with the same visibility rules it would have in the original game.

We call such a conversion procedure PUBLICTEAMCONVERSION, and the related pseudocode is shown in Algorithm 1.

In Appendix A we give the definitions of two mapping function among pure strategies of the original and converted game. We define:

- $\rho : \Pi_{\mathcal{T}} \to \Pi_t$ is the function mapping each team joint pure strategy in the original game into a corresponding pure strategy of the coordinator in the converted game.
- $\sigma : \Pi_t \to \Pi_{\mathcal{T}}$ is the function mapping each pure strategy of the coordinator in the converted game into a corresponding team joint pure strategy in the original game.

It is therefore possible to state the main result of this paper.

Theorem 2. *Given a public-turn-taking vEFG $\mathcal{G}$, and the correspondent converted game $\mathcal{G}' =$ CONVERTGAME$(\mathcal{G})$, a Nash Equilibrium (μ_t, μ_o) in $\mathcal{G}'$ corresponds to a TMECor $(\mu_{\mathcal{T}}, \mu_o)$ in $\mathcal{G}$ where $\mu_{\mathcal{T}} = \sigma(\mu_t)$.*

We show an example of game and results of its conversion in Appendix B, Figure 2 and Figure 3 To simplify the representation, we avoid a full adversarial team game, and instead focus on a cooperative game.

Algorithm 1 Public Team Conversion

function CONVERTGAME($\mathcal{G}$)
  initialize $\mathcal{G}'$ new game
  $\mathcal{N}' \leftarrow \{t, o\}$
  $h'_\varnothing \leftarrow$ PUBTEAMCONV($h_\varnothing, \mathcal{G}, \mathcal{G}'$) ▷ new game root
  return $\mathcal{G}'$

function PUBTEAMCONV(h, $\mathcal{G}$, $\mathcal{G}'$)
  initialize $h' \in \mathcal{H}'$
  if $h \in \mathcal{Z}$ **then** ▷ terminal node
    $h' \leftarrow h' \in \mathcal{Z}'$
    $u'_p(h') \leftarrow u_p(h) \quad \forall p \in \mathcal{N}$
  else if $\mathcal{P}(h) \in \{o, c\}$ **then** ▷ opponent or chance
    $\mathcal{P}'(h') \leftarrow \mathcal{P}(h)$
    $\mathcal{A}'(h') \leftarrow \mathcal{A}(h)$
    if h is chance node **then**
      $\sigma'_c(h') = \sigma_c(h)$
    for $a' \in \mathcal{A}'(h')$ **do**
      $Pub'_t(a') \leftarrow$ check $Pub_{\mathcal{T}}(a') = \text{pub}$
      $Pub'_o(a') \leftarrow Pub_o(a')$
      $h'a' \leftarrow$ PUBTEAMCONV(ha', $\mathcal{G}$, $\mathcal{G}'$)
  else ▷ team member
    $\mathcal{P}'(h') = t$
    $\mathcal{A}'(h') \leftarrow \times_{I \in \mathcal{S}_{\mathcal{T}}(h)} \mathcal{A}(I)$ ▷ prescriptions
    for $\Gamma' \in \mathcal{A}'(h')$ **do**
      $Pub'_t(\Gamma') \leftarrow \text{seen}, Pub'_o(\Gamma') \leftarrow \text{unseen}$
      $a' \leftarrow \Gamma'[\mathcal{I}(h)]$ ▷ extract chosen action
      initialize $h'' \in \mathcal{H}'$
      $\mathcal{A}'(h'') \leftarrow \{a'\}$
      $\mathcal{P}(h'') = c$
      $Pub'_t(a') \leftarrow \text{seen}$
      $Pub'_o(a') = Pub_o(a')$
      $\sigma'_c(h'') =$ play a' with probability 1
      $h''a' \leftarrow$ PUBTEAMCONV(ha', $\mathcal{G}$, $\mathcal{G}'$)
      $h'\Gamma \leftarrow h''$
  return h'

For notational clarity, we use $\leftarrow$ to indicate the assignment of a value to a function or node in the converted game. This assignment will update the data structures in $\mathcal{G}'$ accordingly

Pruning techniques

The general procedure presented in the previous section allows us to prove the theoretical soundness of our approach in the general setting. However, this formalization is not suited for practical applications, because the prescriptions of the coordinator produce a large fanout of the game tree, since in a public state in which A actions are available for S possible private states we have A^S prescriptions.

While the computation of a TMECor has a unavoidable exponential complexity due to the NP-hardness of the problem, any 2p0s solving algorithm can strongly benefit of any improvement made to the game size.

In this section we describe three pruning techniques to attenuate the computational challenges of the converted game.

Pruned Representation. Whenever a prescription is given to a team member that has to play a public action, the played action can be used to exclude the private states for which a different action has been given. This exclusion allows to safely consider only a subset of the possible private state for the public state. In a game in which many actions are played by the same team player, the progressive reduction of possible private states can effectively lower the exponential fanout. This pruning effect is particularly effective when the chosen prescriptions are various, suggesting many different actions; on the other hand, non informative prescriptions associating the same action to all private states do not allow the exclusion of any private state.

Folding Representation. In the original game, chance outcomes are explicitly represented independently by the visibility of the outcomes, branching the game tree in different subgames according to the specific outcome. Consider the case of the extraction of a specific outcome when this is a private information for a team member and unseen by the adversary. In the converted game, such a outcome is not visible to any player, and can therefore be safely postponed as long as no specific action depends on it. The folding representation takes advantage of this property to avoid sampling these types of private state, and instead sampling an action from the prescription depending on the probability (also called belief) that in a certain point in the game a specific private state is present given the previous actions of all players. The dummy chance nodes h'' instantiated in Algorithm 1 therefore may present different actions each with a probability that is the sum of the probability of the private states for which that action has been prescribed. This representation also takes advantage of the computed beliefs to reduce the prescriptions given by the coordinator as in the pruning representation.

Overall, the game size is reduced in case many private states for the players are sampled. Such a representation is similar to the public belief state representation proposed in (Brown et al. 2020). The name folding representation derives from the fact that trajectories with same public actions but different private states are folded one over the other in the converted game.

Safe Imperfect Recallness. Whenever a team player is in a state with three or more actions available, a specific action is played and some possible private states are excluded from the belief. The specific actions prescribed for the excluded states are not important to describe the information state of the player; we can therefore forget part of the prescription and have imperfect recall among different prescriptions with different prescribed actions for excluded states. For example, given 2 player states $\{0, 1\}$ and 3 actions $\{A, B, C\}$, consider prescriptions AB and AC. In the case in which we observe action A after them, there is no difference between having chosen one of the prescriptions over the other; indeed the result is that action A has been played and the player knows to have private state 0.

While this pruning technique does not directly reduce the number of nodes, it reduces the number of information sets, simplifying the information structure of the game. This reduces the space requirements to represent the strategies, and makes the algorithms converge faster. This pruning tech-

nique is theoretically sound, and the convergence properties of CFR in this imperfect recall setting have already been addressed by (Lanctot et al. 2012). For a graphical representation of the effects of the pruning and folding techniques, we can check their effects on Figures 4 and 5 of Appendix B.

Experiments

In this section, we evaluate the benefits of the pruning techniques when applied to a toy game, and present the application of our conversion procedure to multiplayer instances of Kuhn and Leduc poker.

Impact of Pruning techniques

To evaluate the impact of pruning techniques, we designed a simple parametric game for which we are able to determine in closed-form the total number of nodes.

Suppose a two player game $\mathcal{G}$ in which there are C chance outcomes at root, observed by P1 and not by P2, followed by H levels of actions of P1, each with A actions each and one last level of P2. P2 has no visibility of any of the previous actions, thus all his nodes are in the same information set.

We evaluate the size of four different representations, for varying parameters A, C, H:

- **Normal form plans**: as a baseline comparison, we compute the number of normal form plans for P1 in the game;
- **Basic representation**: we compute the total number of nodes of the coordinator player generated by Algorithm 1 from P1 nodes. We do not consider the dummy chance nodes since those nodes can easily be avoided in a practical implementation by automatically applying their action every time a prescription is played;
- **Pruning representation**: we compute the total number of nodes of the coordinator player from P1 nodes, considering a lower fanout due to progressive pruning of private state. Similarly as for the basic representation, we do not consider dummy chance nodes;
- **Folding representation** we compute the total number of nodes of the coordinator player from P1 nodes, considering one node for each public state as in the folding representation. In this case, chance nodes cannot be avoided as multiple outcomes may be available. We therefore consider an extra node for each coordinator's prescription.

The evaluation is performed for a varying number of levels H. Similar considerations have also been done for a variant of the game in which we have two turn of chance plays at the root of the game, in which one out of C outcomes respectively for P1 and P2 are sampled. This variation in the private information structure corresponds to the one present in Kuhn poker. The derivation of the total number of nodes for each representation is presented in Appendix C. Table 1 presents the total results in case of $C = 3$ and $A = 2$. We observe that pruning technique is particularly effective at dampening the exponential factor due to the combinatorial structure of prescriptions, thanks to the belief-based pruning.

Moreover, folding technique combines the benefits of the pruning technique, with the specific tradeoff imposed by the use of delayed chances. indeed, all the histories belonging to the same public state are represented in a single node, at the cost of a chance node added after each prescription. This tradeoff is useful for game states in which many private states are possible. On the other hand, when the possible private states are reduced to one or two, the extra chance nodes added are increasing the size of the game tree more than in the pruning representation. We decide to use the folding representation as our focus in on poker instances with an information structure similar to Table 1b.

Application to team poker

To test the convergence to a TMECor of our approach in a real case, we apply our conversion procedure to the multiplayer versions of Kuhn poker and Leduc poker. The Kuhn instances we use are parametric in the number of ranks available, and on whether the adversary plays first, second or third in the game. Similarly, Leduc instances are parametric on the number of ranks, on the position of the adversary, but also on the number of raises that can be made.

We directly implemented the converted games in Open Spiel (Lanctot et al. 2019), and then took advantage of the already available 2p0s solvers. More details regarding the experimental setting are available in Appendix D, while Appendix E shows the sizes of the converted games for a varying number of ranks and raises.

Figure 1a and Figure 1b show the convergence performances on the converted games of Linear CFR+ (Brown and Sandholm 2019a). We report a paired plot showing both the value and exploitability convergence, along with the optimal value of a TMECor as computed by (Farina et al. 2021), represented as horizontal dashed lines in the team value plot. The results are coherent with the theoretical result of Theorem 2; the Nash Equilibrium returned by the algorithm has the same value of the TMECor of the original game.

Conclusions

This paper presented a conversion procedure to transform an adversarial team game in a two player zero sum game, such that a Nash Equilibrium in the converted game corresponds to a TMECor in the original game. This conversion procedure is based on the use of public information among team members, which allows to remove each player's private information in the converted representation at the cost of a exponential increase in the game size. This is unavoidable due to the NP-hardness of the problem of computing a TMECor in a generic team game, but some pruning techniques are proposed to obtain a resulting which is more computationally tractable. We proved our approach both theoretically and empirically on Kuhn and Leduc poker.

This conversion retains the public structure of the original game, and will allow the application of more scalable techniques for the resolution of adversarial team games, such as abstractions and continual resolving. Future works leveraging our techniques in the converted game have therefore the opportunity to overcome current state of the art techniques on larger game instances. Another direction is the development of a conversion procedure to produce a pruned or folded representation in online fashion, avoiding the construction of the entire tree.

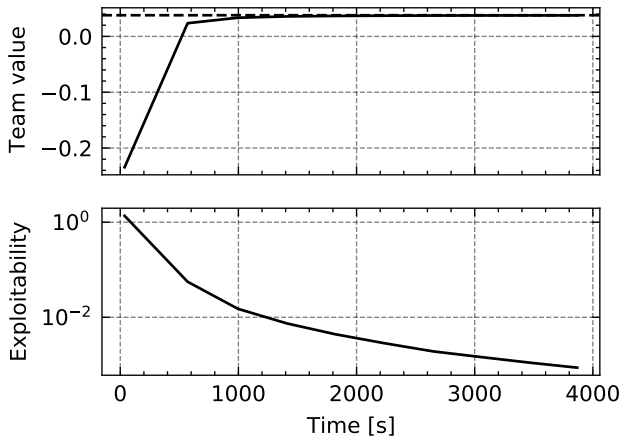
H	normal	basic	pruning	folding
1	8.00E+00	3.00E+00	3.00E+00	9.00E+00
2	6.40E+01	2.70E+01	2.70E+01	7.50E+01
3	5.12E+02	2.19E+02	1.35E+02	3.75E+02
4	4.10E+03	1.76E+03	5.19E+02	1.46E+03
5	3.28E+04	1.40E+04	1.72E+03	4.86E+03
6	2.62E+05	1.12E+05	5.18E+03	1.48E+04
7	2.10E+06	8.99E+05	1.46E+04	4.18E+04
8	1.68E+07	7.19E+06	3.92E+04	1.13E+05
9	1.34E+08	5.75E+07	1.01E+05	2.93E+05
10	1.07E+09	4.60E+08	2.55E+05	7.40E+05
11	8.59E+09	3.68E+09	6.27E+05	1.82E+06
12	6.87E+10	2.95E+10	1.51E+06	4.41E+06
13	5.50E+11	2.36E+11	3.59E+06	1.05E+07
14	4.40E+12	1.88E+12	8.40E+06	2.46E+07

(a) case in which only P1 has private information

H	normal	basic	pruning	folding
1	8.00E+00	9.00E+00	9.00E+00	9.00E+00
2	6.40E+01	8.10E+01	8.10E+01	7.50E+01
3	5.12E+02	6.57E+02	4.05E+02	3.75E+02
4	4.10E+03	5.26E+03	1.56E+03	1.46E+03
5	3.28E+04	4.21E+04	5.16E+03	4.86E+03
6	2.62E+05	3.37E+05	1.55E+04	1.48E+04
7	2.10E+06	2.70E+06	4.37E+04	4.18E+04
8	1.68E+07	2.16E+07	1.17E+05	1.13E+05
9	1.34E+08	1.73E+08	3.04E+05	2.93E+05
10	1.07E+09	1.38E+09	7.65E+05	7.40E+05
11	8.59E+09	1.10E+10	1.88E+06	1.82E+06
12	6.87E+10	8.84E+10	4.53E+06	4.41E+06
13	5.50E+11	7.07E+11	1.08E+07	1.05E+07
14	4.40E+12	5.65E+12	2.52E+07	2.46E+07

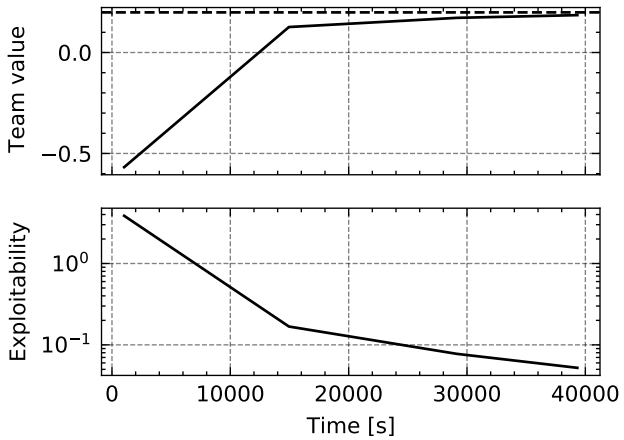
(b) case in which both P1 and P2 have private information

Table 1: Comparison of total number of nodes for $C = 3$, $A = 2$.



Peak RAM occupation: 96 MB

(a) 3-player Kuhn poker, with 4 card ranks and adversary playing first.



Peak RAM occupation: 912 MB

(b) 3-player Leduc poker, with 3 card ranks, 1 raise, and adversary playing first.

Figure 1: Performances of Linear CFR+ applied to Poker game instances.

References

Brown, N., and Sandholm, T. 2017. Libratus: The superhuman AI for no-limit poker. In Sierra, C., ed., *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, 5226–5228. ijcai.org.

Brown, N., and Sandholm, T. 2019a. Solving imperfect-information games via discounted regret minimization. In *AAAI*.

Brown, N., and Sandholm, T. 2019b. Superhuman ai for multiplayer poker. *Science* 365(6456):885–890.

Brown, N.; Bakhtin, A.; Lerer, A.; and Gong, Q. 2020. Combining deep reinforcement learning and search for imperfect-information games. *arXiv:2007.13544 [cs]*. arXiv: 2007.13544.

Brown, G. 1951. Iterative solution of games by fictitious play. *Activity Analysis of Production and Allocation* 13.

Cacciamani, F.; Celli, A.; Ciccone, M.; and Gatti, N. 2021. Multi-agent coordination in adversarial environments through signal mediated strategies. In *AAMAS*.

Celli, A., and Gatti, N. 2018. Computational results for extensive-form adversarial team games. In *AAAI*.

Celli, A.; Ciccone, M.; Bongo, R.; and Gatti, N. 2019. Coordination in adversarial sequential team games via multi-agent deep reinforcement learning. *ArXiv* abs/1912.07712.

Farina, G.; Celli, A.; Gatti, N.; and Sandholm, T. 2018. Ex ante coordination and collusion in zero-sum multi-player extensive-form games. In *NeurIPS*.

Farina, G.; Celli, A.; Gatti, N.; and Sandholm, T. 2021. Connecting optimal ex-ante collusion in teams to extensive-

form correlation: Faster algorithms and positive complexity results. In *ICML*.

Foerster, J. N.; Song, H. F.; Hughes, E.; Burch, N.; Dunning, I.; Whiteson, S.; Botvinick, M.; and Bowling, M. H. 2019. Bayesian action decoder for deep multi-agent reinforcement learning. *ArXiv* abs/1811.01458.

Kovařík, V.; Schmid, M.; Burch, N.; Bowling, M.; and Lisý, V. 2020. Rethinking formal models of partially observable multiagent decision making. *arXiv:1906.11110 [cs]*. arXiv: 1906.11110.

Lanctot, M.; Gibson, R.; Burch, N.; Zinkevich, M.; and Bowling, M. 2012. No-regret learning in extensive-form games with imperfect recall. *arXiv:1205.0622 [cs]*. arXiv: 1205.0622.

Lanctot, M.; Lockhart, E.; Lespiau, J.-B.; Zambaldi, V.; Upadhyay, S.; Pérolat, J.; Srinivasan, S.; Timbers, F.; Tuyls, K.; Omidshafiei, S.; et al. 2019. Openspiel: A framework for reinforcement learning in games. *arXiv preprint arXiv:1908.09453*.

Moravcík, M.; Schmid, M.; Burch, N.; Lisý, V.; Morrill, D.; Bard, N.; Davis, T.; Waugh, K.; Johanson, M. B.; and Bowling, M. H. 2017. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science* 356:508 – 513.

Nayyar, A.; Mahajan, A.; and Teneketzis, D. 2013. Decentralized stochastic control with partial history sharing: A common information approach. *IEEE Transactions on Automatic Control* 58:1644–1658.

Sokota, S.; Lockhart, E.; Timbers, F.; Davoodi, E.; D'Orazio, R.; Burch, N.; Schmid, M.; Bowling, M. H.; and Lanctot, M. 2021. Solving common-payoff games with approximate policy iteration. In *AAAI*.

Vinyals, O.; Babuschkin, I.; Czarnecki, W. M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D. H.; Powell, R.; Ewalds, T.; Georgiev, P.; Oh, J.; Horgan, D.; Kroiss, M.; Danihelka, I.; Huang, A.; Sifre, L.; Cai, T.; Agapiou, J.; Jaderberg, M.; Vezhnevets, A.; Leblond, R.; Pohlen, T.; Dalibard, V.; Budden, D.; Sulsky, Y.; Molloy, J.; Paine, T.; Gulcehre, C.; Wang, Z.; Pfaff, T.; Wu, Y.; Ring, R.; Yogatama, D.; Wünsch, D.; McKinney, K.; Smith, O.; Schaul, T.; Lillicrap, T.; Kavukcuoglu, K.; Hassabis, D.; Apps, C.; and Silver, D. 2019. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature* 1–5.

Von Stengel, B., and Koller, D. 1997. Team-maxmin equilibria. *Games and Economic Behavior* 21(1):309 – 321.

A Proofs

Lemma 1 (Transformation into public turn-taking game)**.** *Any vEFG can be made public turn-taking by adding player nodes with a single noop action. This modification will not increase the size of the game by a factor larger than* $(|\mathcal{N}|+1)|\mathcal{H}|^2$.

Sketch of Proof. A very simple procedure that allows to prove the lemma is the following: we can set for each level of the converted game a player, by cycling through all players (chance included). Then we can add all the histories of the original game one by one, while imposing that at each level only the chosen player can play. If the history has no action assigned to the level's player, then we can add a dummy player node, with only a noop operation, and try to prosecute with the actions of the original history in the next node. The visibility of the noop actions is unseen for all players apart from the one playing.

This procedure guarantees to get a strategically equivalent game by adding at most $\mathcal{O}((|\mathcal{N}|+1)|\mathcal{H}|)$ for any of the $|\mathcal{H}|$ histories in the original game. This proves that the number of histories in the converted game is $\mathcal{O}((|\mathcal{N}|+1)|\mathcal{H}|^2)$ □

Lemma 3. *Given a public-turn-taking vEFG* $\mathcal{G}$*, and the correspondent converted game* $\mathcal{G}' = \textsc{ConvertGame}(\mathcal{G})$*, each joint pure strategy* $\pi_{\mathcal{T}}$ *in* $\mathcal{G}$ *can be mapped to a strategy* π_t *in* $\mathcal{G}'$*, such that the traversed histories have been mapped by* PUBLICTEAMCONVERSION*. Formally:*

$$\forall \pi_{\mathcal{T}}\ \exists \pi_t\ \forall \pi_o, \pi_c: \quad (\textsc{PubTeamConversion}(h))_{h \text{ reached by playing } (\pi_{\mathcal{T}}, \pi_o, \pi_c) \text{ in } \mathcal{G}} \equiv (h')_{h' \text{ reached by playing } (\pi_t, \pi_o, \pi_c) \text{ in } \mathcal{G}'}$$

Proof. We can prove Lemma 3 recursively by traversing both $\mathcal{G}$ and $\mathcal{G}'$ while constructing the equivalent pure strategy in the converted game. We start by $h_\varnothing$ and $h'_\varnothing$. We know that $h'_\varnothing = \textsc{PublicTeamConversion}(h_\varnothing)$.

Let h and h' be the nodes currently reached recursively in $\mathcal{G}$ and $\mathcal{G}'$, such that $h' = \textsc{PublicTeamConversion}(h)$, with the guarantee that correspondent histories in the trajectories traversed up to this point in the two games have such a property. We thus have the guarantee that h and h' are both terminal or both share the same player. Then:

- Case **team member node**
  Let $a = \pi_{\mathcal{T}}[\mathcal{I}(h)]$ be the action specified by $\pi_{\mathcal{T}}$ to be taken at $\mathcal{I}(h)$. We can construct a prescription $\Gamma = (\pi_{\mathcal{T}}[I])_{I \in \mathcal{S}[h]}$ equivalent to the pure strategy $\pi_{\mathcal{T}}$ in this public state. We set $\pi_t[\mathcal{I}'(h')] = \Gamma$, and prosecute our proof from the two reached nodes $h'\Gamma a$ and ha. The construction procedure PUBLICTEAMCONVERSION guarantees In fact that $h'\Gamma a = \textsc{PubTeamConversion}(ha)$.
- Case **chance or opponent node**
  π_o and π_c are common to both the traversals. This guarantees that the action a suggested by the policy is equal, and by construction of the conversion procedure $h'a' = \textsc{PubTeamConversion}(ha)$. We can thus proceed with the proof considering $h'a$ and ha.
- Case **terminal node**
  By construction, they have the same value for all players. This concludes the recursive proof.

□

Lemma 4. *Given a public-turn-taking vEFG* $\mathcal{G}$*, and the correspondent converted game* $\mathcal{G}' = \textsc{ConvertGame}(\mathcal{G})$*, each coordinator pure strategy* π_t *in* $\mathcal{G}'$ *can be mapped to a strategy* $\pi_{\mathcal{T}}$ *in* $\mathcal{G}$*, such that the traversed histories have been mapped by* PUBLICTEAMCONVERSION*. Formally:*

$$\forall \pi_t\ \exists \pi_{\mathcal{T}}\ \forall \pi_o, \pi_c: \quad (\textsc{PubTeamConversion}(h))_{h \text{ reached by playing } (\pi_{\mathcal{T}}, \pi_o, \pi_c) \text{ in } \mathcal{G}} \equiv (h')_{h' \text{ reached by playing } (\pi_t, \pi_o, \pi_c) \text{ in } \mathcal{G}'}$$

Proof. We can prove Lemma 4 recursively by traversing both $\mathcal{G}'$ and $\mathcal{G}$ while constructing the equivalent pure strategy in the original game. We start by $h'_\varnothing$ and $h_\varnothing$. We know that $h'_\varnothing = \textsc{PublicTeamConversion}(h_\varnothing)$.

Let h' and h be the nodes currently reached recursively in $\mathcal{G}'$ and $\mathcal{G}$, such that $h' = \textsc{PublicTeamConversion}(h)$, and with the guarantee that correspondent histories in the trajectories traversed in the two games have such a property. We thus have the guarantee that h and h' are both terminal or both share the same player. Then:

- Case **team member node**
  Let $\Gamma = \pi_t[\mathcal{I}(h')]$ be the prescription specified by π_t to be taken at $\mathcal{I}'(h')$. We can extract the prescribed action $a = \Gamma[I]$ to be played in history h. We set $\pi_{\mathcal{T}}[\mathcal{I}(h)] = a$, and prosecute our proof from the two reached nodes $h'\Gamma a$ and ha. The PUBLICTEAMCONVERSION procedure guarantees In fact that $h'\Gamma a = \textsc{PubTeamConversion}(ha)$.

- Case **chance or opponent node**
  π_o and π_c are common to both the traversals. This guarantees that the action a suggested by the policy is equal, and by construction of the conversion procedure $h'a' = \text{PUBTEAMCONVERSION}(ha)$. We can thus proceed with the proof considering $h'a$ and ha.
- Case **terminal node**
  By construction, they have the same value for all players. This concludes the recursive proof.

□

Definition 2 (Mapping functions among original and converted game)**.** *We define:*

- $\rho : \Pi_{\mathcal{T}} \to \Pi_t$ *is the function mapping each* $\pi_{\mathcal{T}}$ *to the* π_t *specified by the procedure described in the proof of Lemma 3.*
- $\sigma : \Pi_t \to \Pi_{\mathcal{T}}$ *is the function mapping each* π_t *to the* $\pi_{\mathcal{T}}$ *specified by the procedure described in the proof of Lemma 4.*

Those two functions can also be extended to mixed strategies, by converting each pure plan and summing the probability masses of the converted plans. Formally:

$$\forall \mu_{\mathcal{T}} \in \Delta^{\Pi_{\mathcal{T}}} : \rho(\mu_{\mathcal{T}})[\pi_t] = \sum_{\pi_{\mathcal{T}} : \rho(\pi_{\mathcal{T}}) = \pi_t} \mu_{\mathcal{T}}(\pi_{\mathcal{T}})$$
$$\forall \mu_t \in \Delta^{\Pi_t} : \sigma(\mu_t)[\pi_{\mathcal{T}}] = \sum_{\pi_t : \sigma(\pi_t) = \pi_{\mathcal{T}}} \mu_t(\pi_t)$$

Corollary 4.1 (Payoff equivalence)**.** *A public-turn-taking vEFG* $\mathcal{G}$ *and* $\mathcal{G}' = \text{CONVERTGAME}(\mathcal{G})$ *are payoff-equivalent. Formally:*

$$\forall \pi_{\mathcal{T}}\ \forall \pi_o, \pi_c : u_{\mathcal{T}}(\pi_{\mathcal{T}}, \pi_o, \pi_c) = u_t(\rho(\pi_{\mathcal{T}}), \pi_o, \pi_c)$$
$$\forall \pi_t\ \forall \pi_o, \pi_c : u_{\mathcal{T}}(\sigma(\pi_t), \pi_o, \pi_c) = u_t(\pi_t, \pi_o, \pi_c)$$

We can now prove the main result of the present work.

Theorem 2. *Given a public-turn-taking vEFG* $\mathcal{G}$*, and the correspondent converted game* $\mathcal{G}' = \text{CONVERTGAME}(\mathcal{G})$*, a Nash Equilibrium* (μ_t, μ_o) *in* $\mathcal{G}'$ *corresponds to a TMECor* $(\mu_{\mathcal{T}}, \mu_o)$ *in* $\mathcal{G}$ *where* $\mu_{\mathcal{T}} = \sigma(\mu_t)$*.*

Proof. By hypothesis we have that:

$$\mu_t^* \in \arg\max_{\mu_t \in \Delta^{\Pi_t}} \min_{\mu_o \in \Delta^{\Pi_o}} \sum_{\substack{\pi_t \in \Pi_t \\ \pi_o \in \Pi_o \\ \pi_c \in \Pi_c}} \mu_t(\pi_t)\mu_o(\pi_o)\mu_c(\pi_c)u_t(\pi_t, \pi_o, \pi_c)$$

We need to prove:

$$\sigma(\mu_t^*) \in \arg\max_{\mu_{\mathcal{T}} \in \Delta^{\Pi_{\mathcal{T}}}} \min_{\mu_o \in \Delta^{\Pi_o}} \sum_{\substack{\pi_{\mathcal{T}} \in \Pi_{\mathcal{T}} \\ \pi_o \in \Pi_o \\ \pi_c \in \Pi_c}} \mu_{\mathcal{T}}(\pi_{\mathcal{T}})\mu_o(\pi_o)\mu_c(\pi_c)u_{\mathcal{T}}(\pi_{\mathcal{T}}, \pi_o, \pi_c)$$

Let $\min_{TMECor}(\mu_{\mathcal{T}})$ and $\min_{NE}(\mu_t)$ be the inner minimization problem in the TMECor and NE definition respectively.
Absurd. Suppose $\exists\ \bar{\mu}_{\mathcal{T}}$ with a greater value than $\sigma(\mu_t^*)$. Formally:

$$\min_{TMECor}(\bar{\mu}_{\mathcal{T}}) > \min_{TMECor}(\mu_t^*)$$

In such a case, we could define $\bar{\mu}_t = \rho(\bar{\mu}_{\mathcal{T}})$ having value:

$$\min_{NE}(\bar{\mu}_t) = \min_{TMECor}(\bar{\mu}_{\mathcal{T}}) > \min_{TMECor}(\sigma(\mu_t^*)) = \min_{NE}(\mu_t^*)$$

where the equalities are due to the payoff equivalence. However this is absurd since by hypothesis μ_t^* is a maximum. Therefore necessarily:

$$\sigma(\mu_t^*) \in \arg\max_{\mu_{\mathcal{T}} \in \Delta^{\Pi_{\mathcal{T}}}} \min_{NE}(\mu_{\mathcal{T}})$$

□

B Converted game representation

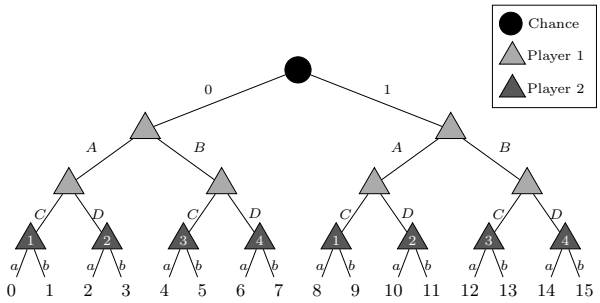


Figure 2: Example of a cooperative game. Player 2 can see all actions apart from chance outcomes 0, 1. Nodes of a player with same number are in the same infoset.

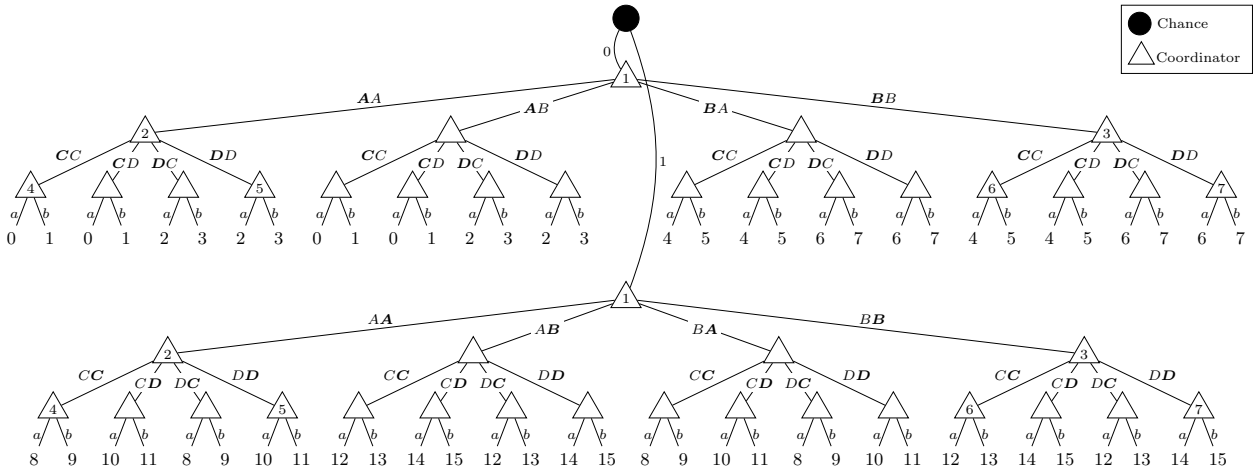


Figure 3: Example of Figure 2 converted. Nodes of a player with same number are in the same infoset. For notational clarity, dummy chance nodes are not represented, prescriptions list the action to take for private state 0 and 1, the action taken afterward is in bold in the prescription.

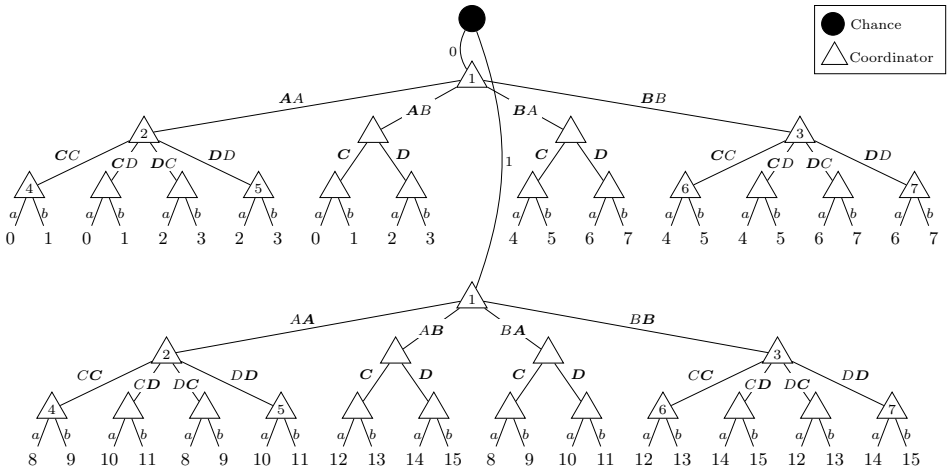


Figure 4: Example of Figure 2 converted using belief pruning. Nodes of a player with same number are in the same infoset. For notational clarity, dummy chance nodes are not represented, prescriptions list the action to take for private state 0 and 1, the action taken afterward is in bold in the prescription.

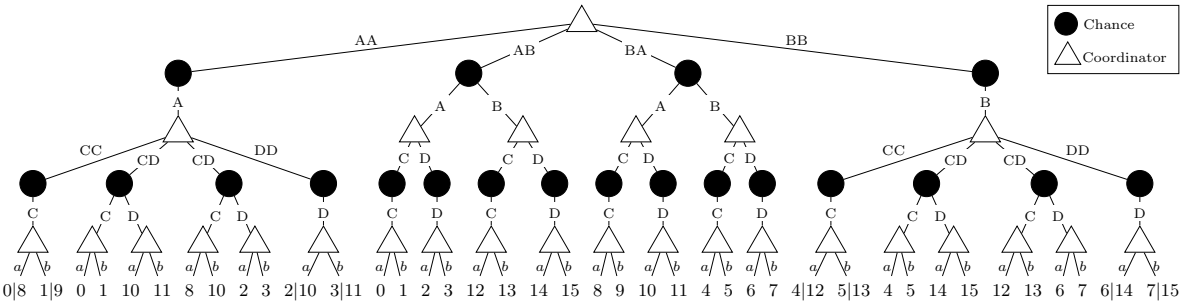


Figure 5: Example of Figure 2 converted using folded representation. For notational clarity, prescriptions list the action to take for private state 0 and 1. Terminal nodes in the form $x|y$ represent a terminal node which has a weighted average value with respect to the outcomes x and y.

C Pruning evaluation

We proceed to analyze the size in total number of coordinator nodes of the converted game for player P1 nodes, depending on the pruning technique used. To formalize the total number of nodes, we use a succession notation, where $s_l(c)$ indicates the number of nodes at level l in which P1 may be in exactly c private states. Such a notation is particularly useful to represent the relation between private states, pruning, and total number of nodes.

Normal form representation. As a baseline comparison, we compute the number of normal form plans in the game. The total number of plans for P1 can be computed as $A^{C \cdot H}$.

Basic Representation. Each of the H level has A^C actions, and we have C independent trees due to the initial chance. Since we do not perform any belief pruning, all nodes have C possible private states.

The correspondent succession is:

$$x_0(c) = \begin{cases} 0 & \text{if } c \neq C \\ C & \text{if } c = C \end{cases} \quad \text{initial C chance outcomes}$$

$$x_l(c) = \begin{cases} 0 & \text{if } c \neq C \\ x_{l-1}(c) \cdot A^C & \text{if } c = C \end{cases} \quad A^C \text{ fanout at each level}$$

Therefore:

$$\begin{aligned} \text{tot}(C, A, H) &= \sum_{l=0}^{H} \sum_{c=1}^{C} x_l(c) \\ &= C \cdot \sum_{i=1}^{H} (A^C)^i \end{aligned}$$

Pruning Representation. Given a node with a generic number I of private states, we can work by induction to retrieve the number of generated nodes:

- children left with I possible infostates: A. They correspond to the nodes reached through a prescription assigning the same action for all I states;
- children left with $I-1$ possible infostates: $A \cdot (A-1)^1 \cdot (I-1)$. They correspond to the nodes reached through a prescription assigning any of the A actions to the state corresponding to the card drawn in this subtree (defined by the chance outcomes) and to other $I-2$ states, and assigning any of the remaining $A-1$ actions to the remaining state;
- children left with $I-2$ possible infostates: $A \cdot (A-1)^2 \cdot \binom{I-1}{2}$. They correspond to the nodes reached through a prescription assigning any of the A actions to the state corresponding to the card drawn in this subtree (defined by the chance outcomes) and to other $I-3$ states, and assigning any of the remaining $A-1$ actions to the 2 remaining states;
- ...
- children left with 1 possible infostates: $A \cdot (A-1)^{C-1} \cdot \binom{C-1}{C-1}$. They correspond to the nodes reached through a prescription assigning any of the A actions to the state corresponding to the card drawn in this subtree (defined by the chance outcomes), and assigning any of the remaining $A-1$ actions to the $I-1$ remaining states.

We can generalize this pattern. Children left with i possible private states out of available I:

$$n(i, I) = A \cdot (A-1)^{I-i} \cdot \binom{I-1}{I-i} \text{ for } i \in [1, I]$$

As a check:

$$\begin{aligned}\sum_{i=1}^{C} n(i, I) &= \sum_{i=1}^{I} A \cdot (A-1)^{I-i} \cdot \binom{I-1}{I-i} = \\ &= A \sum_{j=0}^{I-1} \binom{I-1}{j} (A-1)^j 1^{I-1-j} = \\ &= A \cdot [(A-1)+1]^{I-1} = A^I\end{aligned}$$

which corresponds to the expected A^I prescriptions available in the current node.

Then repartition of each level's nodes will depend on the number of nodes having a certain number of private state in the previous state, according to the repartition indicated by $n(i, I)$. In particular, each of the $b_{l-1}(c)$ nodes will generate $n(i, c)$ children with i private states.

The correspondent succession is:

$$y_0(C) = \begin{cases} 0 & \text{if } c \neq C \\ C & \text{if } c = C \end{cases} \text{ initial C chance outcomes}$$

$$y_l(c) = \sum_{i=c}^{C} b_{l-1}(i) \cdot n(c, i)$$

Note that we do not count auxiliary chance nodes, since in practical implementation they can be easily compacted with the previous coordinator nodes.

Therefore:

$$\text{tot}(C, A, H) = \sum_{l=0}^{H} \sum_{c=1}^{C} y_l(c)$$

Folding Representation. In this representation we have no initial chance sampling, and each coordinator nodes presents a chance node per prescriptions, each with a variable number of children depending on the number of unique actions.

To compute the total number of nodes per level, we can acknowledge that each coordinator node with c private states corresponds to c nodes (all with c private states) in the pruning representation. Therefore, at each level we have a number of coordinator node $z_l(c) = y_l(c)/c$

In this case, chance nodes have to be considered in the total nodes computed, since they cannot be easily reduced. In particular, each coordinator node has associated a chance node per prescription action available.

Therefore:

$$z_l(c) = y_l(c)/c$$

$$\text{tot}(C, A, H) = \sum_{l=0}^{H} \sum_{c=1}^{C} y_l(c)/c \cdot (A^c + 1)$$

Moreover, such an analysis can be extended also to the case of 2 initial level of chance nodes, extracting one out of C private states for P1 and P2 respectively. In this case, basic and pruning representation have a different starting condition, with $x_0(C) = y_0(C) = C^2$, while the folding representation has no changes.

D Experimental settings

Poker instances

We refer to the three player generalizations of Kuhn and Leduc poker proposed by (Farina et al. 2018).

As all poker games, at the start of the game each player antes one to the pot, and receives a private card. Then players play sequentially in turn. Each player may check by adding to the pot the difference between the higher bet made by other players and their current bet (i.e. by matching the maximum bet made by others). Each player may fold whenever a check requires to put more money into the pot and the player instead decides to withdraw. Each player may raise whenever the maximum number of raises allowed by the game is not reached, by adding to the pot the amount required by a check plus an extra amount called raise amount. A betting round ends when all non-folded players except the last raising player have checked.

In **Kuhn poker**, there are three players and k possible ranks with k different ranks. The maximum number of raises is one, and the raising amount is 1. At the end of the first round, the showdown happens. The player having the highest card takes all the pot as payoff.

In **Leduc poker**, there are three players, k possible ranks having each 3 cards in the deck, and 1 or 2 raises. The raise amount is 2 for the first raise and 4 for the second raise. At the end of the first round, a public card is shown, and a new round of betting starts from the same player starting in the first round. At the end, the showdown happens. Winning players are having a private card matching the rank of the public card. If no player forms a pair, then the winning player is the one with the card with the highest rank. In case of multiple winners, the pot is split equally.

Implementation and execution details

We implemented the folded representation of both Kuhn and Leduc taking advantage of the OpenSpiel (Lanctot et al. 2019) framework. The framework allowed us to specify the game as an evolving state object, and provided the standard resolution algorithms for the computation of a Nash Equilibrium in the converted game.

The implementation is in Python3.8 and the experiments have been performed on a machine running Ubuntu 16.04 with a 2x Intel Xeon E5-4610 v2 @ 2.3GH CPU. The implementation is single threaded.

E Converted game size

Number of ranks	3	3	3	4	4	4	5	5	5
Adversary position	0	1	2	0	1	2	0	1	2
Coordinator nodes	222	291	591	1560	2220	7412	8890	13025	66465
Adversary nodes	219	372	288	1996	5416	2656	12425	54040	16560
Terminal nodes	1320	1704	2436	16584	24536	51800	144740	235660	760520
Chance nodes	1129	1405	2461	10913	14641	40977	85521	119001	514681
Chances with one child only	936	1188	2184	5680	7944	25400	29840	43360	218940
Total number of nodes	2890	3772	5776	31053	46813	102845	251576	421726	1358226
Coordinator information sets	86	113	155	392	556	856	1738	2543	4093
Adversary information sets	12	12	12	16	16	16	20	20	20
Time taken for a full traversal	2.0s	2.3s	3.36s	14.7s	18.1s	37.2s	68.6s	125s	447s

Table 2: Converted Kuhn game characteristics for varying parameters.

Number of ranks	3	3	3	4	4	4
Number of raises	1	1	1	2	2	2
Adversary position	0	1	2	0	1	2
Coordinator nodes	84243	117126	232950	57138	66268	76384
Adversary nodes	60543	98034	134196	32790	38622	46758
Terminal nodes	354999	476187	775233	163580	185994	213098
Chance nodes	284200	378928	694132	160395	184065	211437
Chances with one child only	181020	250908	494544	137044	159202	184738
Total number of nodes	783985	1070275	1836511	413903	474949	547677
Coordinator information sets	7184	7232	7316	5624	5632	5650
Adversary information sets	228	228	228	630	630	630
Time taken for a full traversal	332s	322s	686s	220s	255s	183s

Table 3: Converted Leduc game characteristics for varying parameters.