

Perfect Information Monte Carlo with Postponing Reasoning

Jérôme Arjonilla
Université Paris Dauphine - PSL
Paris, France
jerome.arjonilla@hotmail.fr

Abdallah Saffidine
Potassco Solutions
Potsdam, Germany
abdallah.saffidine@gmail.com

Tristan Cazenave
Université Paris Dauphine - PSL
Paris, France
tristan.cazenave@dauphine.psl.eu

Abstract—Imperfect information games, such as Bridge and Skat, present challenges due to state-space explosion and hidden information, posing formidable obstacles for search algorithms. Determinization-based algorithms offer a resolution by sampling hidden information and solving the game in a perfect information setting, facilitating rapid and effective action estimation. However, transitioning to perfect information introduces challenges, notably one called strategy fusion. This research introduces ‘Extended Perfect Information Monte Carlo’ (EPIMC), an online algorithm inspired by the state-of-the-art determinization-based approach Perfect Information Monte Carlo (PIMC). EPIMC enhances the capabilities of PIMC by postponing the perfect information resolution, reducing alleviating issues related to strategy fusion. However, the decision to postpone the leaf evaluator introduces novel considerations, such as the interplay between prior levels of reasoning and the newly deferred resolution. In our empirical analysis, we investigate the performance of EPIMC across a range of games, with a particular focus on those characterized by varying degrees of strategy fusion. Our results demonstrate notable performance enhancements, particularly in games where strategy fusion significantly impacts gameplay. Furthermore, our research contributes to the theoretical foundation of determinization-based algorithms addressing challenges associated with strategy fusion.

Index Terms—Imperfect Information Games, Search Algorithm, Determinization, Strategy Fusion

I. INTRODUCTION

Search algorithms in artificial intelligence have significantly evolved, demonstrating superhuman performance in games such as Chess, Go [1], Poker [2], Skat [3], and Contract Bridge [4]. Perfect information games, like Chess and Go, where all information is available, have been extensively studied, allowing algorithms to surpass human professionals [1], [5], [6]. In contrast, imperfect information games, including Poker, Skat, and Bridge, where some information is hidden, have been less studied, with only a few algorithms capable of outperforming professional human players [2].

In imperfect information games, two commonly used search methods are regret-based approaches, which excel in Poker and are theoretically convergent but slower [7]–[9], and determinization-based methods, considered state-of-the-art in various trick-taking card games, offering scalability but lacking theoretical guarantees [3], [10]–[12]. In recent years, both methods have incorporated neural networks to enhance

performance and facilitate scalability on large games [2], [13]–[16].

Determinization-based algorithms, like PIMC, operate by sampling hidden information based on current knowledge and using a *perfect information leaf evaluator* to predict game outcomes under perfect information assumptions. These algorithms achieve state-of-the-art performance because solving problems with perfect information is inherently simpler than dealing with imperfect information. Despite their state-of-the-art performance, determinization-based algorithms face challenges, notably encountering ‘strategy fusion’ [10], [17]. This challenge arises from the use of the perfect information leaf evaluator, which independently solves each possible world without considering the uncertainties induced by games with imperfect information.

Within determinization-based algorithms, Perfect Information Monte Carlo (PIMC) [10] is particularly susceptible to strategy fusion due to its early usage on the perfect information leaf evaluator in decision-making. Our study addresses this challenge by postponing the leaf evaluator until a depth d , which mitigates the impact of strategy fusion. The act of postponing the perfect information leaf evaluator at a depth of d introduces new considerations, specifically, it prompts the need for alternative strategies to reason from step 1 to step d . We formally define the problem of strategy fusion, demonstrating that, in the worst case, increasing depth d does not exacerbate strategy fusion, and in every case, there exists a depth d that strictly reduces it. For finite games, there is a depth d that eliminates strategy fusion entirely.

Section II covers notation and a detailed explanation of determinization-based algorithms, specifically addressing PIMC and the strategy fusion challenge. Section III introduces Extended PIMC, incorporating our idea of postponing the leaf evaluator at depth d , which operates online without significant initial costs, making it suitable for diverse games or General Game Playing [18]. It is noteworthy that, like other modern determinization algorithms, there exists the potential to integrate neural networks for enhanced performance. Section IV presents the theoretical results of increasing depth d in determinization-based algorithms. Section V showcases experimental findings across various games, highlighting cases with significant strategy fusion where deeper reasoning improves performance beyond other state-of-the-art methods. Section VI

reviews related research, and Section VII summarizes our contributions and future research directions.

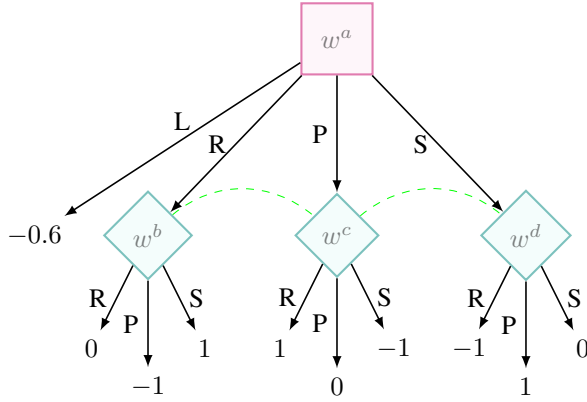


Fig. 1: Variant of ‘Rock-Paper-Scissors’. The red/green square/diamond is the first/second player and the dashed line represents worlds indistinguishable by the second player.

II. NOTATION AND BACKGROUND

A. Notation

Throughout the paper, we use the formalism of Factored-Observation Stochastic Games (FOSG) [19] and utilize Figure 1, a variant of ‘Rock-Paper-Scissors’ to present the notations. Our notation employs subscripts to denote players.

A game, denoted as G , involves \mathcal{N} players, initiating at w^{init} and progressing through successor world states represented by $w \in \mathcal{W}$. Players make joint actions, denoted as $a = (a_1, \dots, a_{\mathcal{N}}) \in \mathcal{A}(w)$, in each world state w , and the game continues until a terminal state is reached. Upon choosing a joint action a , players observe rewards, and the next world state w' is determined probabilistically. During this transition from w to w' , each player receives an observation denoted as $o_i \in \mathcal{O}_i(w, a, w')$, where $\mathcal{O}_i(w, a, w')$ represents the set of possible observations for player i .

In Figure 1, the first player faces the choice between ‘Leave’ with a reward of -0.6 or engaging in the game. If it opts to play, the standard rules apply: Rock beats Scissors, Scissors beats Paper, and Paper beats Rock, with wins yielding 1, losses resulting in -1 , and ties providing 0. The game encompasses four possible world states denoted as $\mathcal{W} = \{w^a, w^b, w^c, w^d\}$, where w^a is the initial state and w^b , w^c , and w^d follow the first action.

The first player has four possible actions in w^a and a null action in other states—specifically, $\mathcal{A}_1(w^a) = \{\text{Leave, Rock, Paper, Scissors}\}$ and $\mathcal{A}_1(w^b) = \mathcal{A}_1(w^c) = \mathcal{A}_1(w^d) = \text{noop}$. The second player has a null action in w^a and three actions in other states—namely, $\mathcal{A}_2(w^a) = \emptyset$ and $\mathcal{A}_2(w^b) = \mathcal{A}_2(w^c) = \mathcal{A}_2(w^d) = \{\text{Rock, Paper, Scissors}\}$.

In this game, the second player receives the observation Play if the first player plays Paper, Scissors, or Rock. Rewards for both players are obtained at the game’s conclusion.

A *history* is a finite sequence of world states and legal actions denoted as $h^t = (w^0, a^0, w^1, a^1, \dots, w^t)$. An *infostate* s_i is a sequence of an agent’s observations and actions, denoted as $s_i^t = (a_i^0, o_i^1, a_i^1, \dots, o_i^{t-1})$. For each history, there exists a unique infostate noted as $s_i(h)$, and for each infostate, there is a set of histories that match the sequence denoted as $\mathcal{H}(s)$. $\mathcal{I}(G)$ represents the set of possible infostates in the game G , and $\Pi(G)$ is the set of possible policies in the game G , where a *policy* π is a function mapping every history h to a probability distribution over actions.

In Figure 1, if we examine the history $h^1 = (w^a, \text{Rock}, w^b)$, considering it from the second player’s perspective, the infostate becomes $s_2^1 = (\text{noop}, \text{Play})$ since no action was taken by the second player, and the observation Play was noted. The set of histories that correspond to s_2^1 includes $\{(w^a, \text{Rock}, w^b); (w^a, \text{Paper}, w^c); (w^a, \text{Scissors}, w^d)\}$.

B. Determinization-based algorithm

Each determinization-based algorithm has its own characteristics, nevertheless, they share some common features such as (i) *samples* a history h according to a probability distribution over the current infostate s ; (ii) uses a *perfect information leaf evaluator* for estimating the value of the sampled world state. In the description of the Algorithm 1 and 2 (i) is noted *Sampling(s)* and (ii) is noted *PerfectAlgo(h)*.

A perfect information leaf evaluator is an algorithm used in games with perfect information to estimate the value of a history h . It predicts the outcome of a game from a specific position. This evaluator can be exhaustive methods like Minimax with Alpha-Beta pruning [20], heuristics methods like Random Rollout (also called *playouts* [21]) or neural network [1], [5], [6].

Determinization-based algorithms are simple and, in practice, achieve great results. Yet, certain problems are encountered such as (i) non-locality and strategy-fusion [10], [17]; (ii) revealing private hidden information [17], [22]; (iii) no theoretical guarantees.

In Figure 1, when conducting sampling from the infostate s_2^1 , the available options include sampling w^b , w^c , or w^d . Utilizing a perfect information leaf evaluator, such as Minimax, on the world state w^b would yield a value of -1 , as the second player optimally plays Paper to maximize their score *i.e.*, minimizes the value of the first player.

C. Strategy fusion

In imperfect information games, histories stemming from the same infostate must be approached with the same strategy,

as players cannot distinguish between them. Formally, $\forall s \in \mathcal{I}(G), \forall h, h' \in \mathcal{H}(s), \pi(\cdot|h) = \pi(\cdot|h')$.

However, determinization-based algorithms deviate from this principle. They employ a perfect information leaf evaluator algorithm to estimate the value at each sampled history. In other words, each history originating from the same infostate is solved using a strategy tailored to that specific history. Formally, $\forall s \in \mathcal{I}(G), \forall h, h' \in \mathcal{H}(s), \pi(\cdot|h) \neq \pi(\cdot|h')$.

In Figure 1, the strategies in w^b , w^c , and w^d must be identical since they originate from the same infostate s_2^1 . For the second player, the optimal strategy results in an average score of 0 (playing with the same probability for all three actions). By back-propagating, the first player opts for Play, leading to an average score of 0, instead of choosing Leave, which results in -0.6 .

From the perspective of a determinization-based algorithm, a policy is tailored to the sampled history. The best policy for the second player is Paper in w^b , Scissors in w^c , and Rock in w^d . In w^b , w^c , and w^d , playing the best policy yields -1 . By back-propagating, the first player concludes that opting for Leave to obtain -0.6 is preferable compared to playing and receiving -1 .

D. Perfect Information Monte Carlo

Perfect Information Monte Carlo (PIMC) is a determinization-based algorithm that is the state-of-the-art of many imperfect information games.

Algorithm 1: PIMC

```

Function PIMC( $s$ ):
  for  $a \in \mathcal{A}(s)$  do
     $score[a] \leftarrow 0$ ;
  end
  while  $budget$  do
     $w \leftarrow \text{Sampling}(s)$ ;
    for  $a \in \mathcal{A}(w)$  do
       $w' \leftarrow w.ApplyMove(a)$ ;
       $score[a] \leftarrow score[a] + PerfectAlgo(w')$ ;
    end
  end
  return Returns the best action on average;

```

PIMC is defined in Algorithm 1 and works as follows (i) samples a world state w according to the probability distribution over the current infostate s ; (ii) plays an action a on the world state w , and observes the next world state w' ; (iii) estimates the world state w' by using the perfect information leaf evaluator; (iv) repeats until the $budget$ is over; (v) selects the action that produces the best results in average.

III. EXTENDED PIMC

As presented, employing the perfect information leaf evaluator results in strategy fusion. In the case of PIMC, this

evaluator is utilized after the first action is played. However, there are no inherent constraints preventing the resolution after the first action, and it is not difficult to believe that postponing the use of the perfect information leaf evaluator could mitigate the issue of strategy fusion.

In the following, we introduce a novel algorithm that embraces this straightforward concept of postponing the leaf evaluator's utilization. The algorithm, termed 'EPIMC' (Extended Perfect Information Monte Carlo), extends the PIMC paradigm by incorporating Extended reasoning at a depth of d , where the instance of $d = 1$ corresponds to PIMC.

Algorithm 2: Extended PIMC

```

Function ExtendedPIMC( $d, s$ ):
  Create the game  $U$  and  $u$ , the initial world state;
  while  $budget$  do
     $w \leftarrow \text{Sampling}(s)$ ;
    Query( $U, u, w, d$ );
  end
  return ImperfectAlgo( $U$ );

Function Query( $U, u, w, d$ ):
  if  $d == 0$  or  $w.IsTerminal()$  then
     $u.value \leftarrow u.value + PerfectAlgo(w)$ ;
    return;
  end
   $w' \leftarrow w.ApplyMove(RandomAction(w))$ ;
  Get  $u'$  associated with  $w'$ ;
  Update dynamics in  $U$  according to  $u'$  and  $u$ ;
  Query( $U, u', w', d - 1$ );

```

The pseudo-code is available in Algorithm 2 and works as follows (i) creates a subgame game U ; (ii) samples a world state w according to the probability distribution over the current infostate s ; (iii) plays an action a on w , observes the next world state w' continue until w' is a terminal node or the depth d is obtained, and updates the dynamic in subgame U ; (iv) estimates the world state w' by using the perfect information leaf evaluator; (v) repeats phases 2 to 4 until the $budget$ is over; (vi) solves the subgame U which have been created during phase 2 to 5 by using an algorithm that does not create strategy fusion (*i.e.* an algorithm that do not use a perfect leaf evaluator to resolve the subgame U).

In the following, a few points are clarified.

Exploration Strategy

In the original PIMC, world state evaluations occur a maximum of $|A|$ times during each sampling phase, with $|A|$ denoting the highest feasible action count. However, applying the same approach at depth d could potentially lead to evaluating $|A|^d$ world states. Therefore, careful consideration of the number of actions per step becomes imperative. In EPIMC, our chosen strategy is to explore one action per sampling iteration, resulting in a singular world state evaluation per sampling instance, however, other choices could have been envisaged.

Depth

While theoretically, any depth d could be employed, practical considerations necessitate a prudent choice of d . Selecting a depth d that is too small can exacerbate strategy fusion issues, as observed in approaches like PIMC. On the other hand, opting for a depth d that is too large demands significant sampling efforts to accurately estimate the subgame U . Moreover, depending on the algorithm employed to solve the subgame U , a larger depth can lead to increased computational costs.

Subgame Resolution

By postponing the application of the perfect information leaf evaluator until a depth of d , alternative reasoning methods must be employed for steps 1 through d . In EPIMC, the subgame U of size d is built to approximate the real game by encapsulating various elements, including infostates, world states, post-action dynamics, and more. In particular, the leaves of the subgame U are average scores obtained from the leaf evaluator.

After the budget is finished, the subgame U is solved using an algorithm that does not create strategy fusion. In other words, one needs an algorithm that works on infostates instead of world states. This choice of algorithm, although potentially resource-intensive, carries a reduced computational burden when applied to a subgame U of size d . One can think of using information set search [23], [24] which operates on infostates according to a minimax rule or CFR/CFR+ [7], [8] that have theoretical guarantees for two players.

IV. THEORETICAL FOUNDATION

In the following, we present the theoretical foundation for determinization-based algorithms that suffer from strategy fusion. We formally (i) define the condition to *create* strategy fusion; (ii) define the quantity of strategy fusion; (iii) prove that, in the worst case, increasing the depth does not increase the strategy fusion, and in every case, there exists a depth d such that the strategy fusion is strictly reduced and in a finite game, there exists a depth d such as the that the strategy fusion is removed.

Definition 1. For any game G , a policy $\pi \in \Pi(G)$ and an infostate $s \in \mathcal{I}(G)$ *creates* strategy fusion if there are $h, h' \in \mathcal{H}(s)$ such that $\pi(h) \neq \pi(h')$. For any game G , a policy $\pi \in \Pi(G)$ *creates* strategy fusion if there is an infostate $s \in \mathcal{I}(G)$ such that π creates strategy fusion in s .

To evaluate the quantity of strategy fusion in $\pi \in \Pi(G)$, we propose the following measure $SF(\pi) = |\{s \text{ such that } \forall s \in S(G), \pi \text{ creates strategy fusion in } s\}|$. In other, we count the number of infostate that create strategy fusion. $SF(\pi) = 0$ implies that there is no strategy fusion.

In the subsequent discussions, for the sake of simplicity, we presume adherence to the policy provided by EPIMC, denoted as $\Pi^{E^d}(G)$ when executing EPIMC at a depth of d . Although alternative choices could have been considered, opting for the

EPIMC policy is more straightforward, as influenced by the uniform growth of the EPIMC policy across the entire depth d space.

Proposition 1. $\forall G, \forall d \in [0, T - 1]$ where $T = \{T \text{ if } G \text{ has a finite horizon } T; \text{ else } \infty\}, \forall \pi \in \Pi^{E^d}(G)$, then $\forall \pi' \in \Pi^{E^{d+1}}(G)$, $SF(\pi, G) \geq SF(\pi', G)$

Proof. For any $s \in \mathcal{I}(G)$, s does not create strategy fusion in $\pi^{1:d}$. Increasing the depth by 1, extends the non-inducing region. Two possibilities arise: (i) if at least one s at $d + 1$ create strategy fusion, it can no longer do so, i.e., $SF(\pi, G) > SF(\pi', G)$; (ii) if all s at $d + 1$ do not create strategy fusion, increasing the depth does not reduce SF, i.e., $SF(\pi, G) \geq SF(\pi', G)$. We conclude $SF(\pi, G) \geq SF(\pi', G)$. \square

Proposition 2. $\forall G, \forall d \in [0, T]$ where $T = \{T \text{ if } G \text{ has a finite horizon } T; \text{ else } \infty\}, \forall \pi \in \Pi^{E^d}(G)$, if $SF(\pi, G) > 0$, then $\exists d' \in [1, T - d], \forall \pi' \in \Pi^{E^{d+d'}}(G)$ such that $SF(\pi, G) > SF(\pi', G)$.

Proof. Leveraging the rationale from Proposition 1, increasing the depth by d' extends the non-inducing region. Given the presence of strategy fusion ($SF(\pi, G) > 0$), at least one infostate creates strategy fusion. Extending the reasoning up to this infostate, located d' away from the original depth d , effectively diminishes strategy fusion. Hence, we establish $SF(\pi, G) > SF(\pi', G)$. \square

We extend Proposition 2 by showing that, in a finite game, there is a depth at which there is no longer strategy fusion.

Proposition 3. $\forall G$ such that G has a finite horizon T , $\forall d \in [1, T - 1], \forall \pi \in \Pi^{E^d}G$ if $SF(\pi, G) > 0$ then $\exists d' \in [1, T - d], \forall \pi' \in \Pi^{E^{d+d'}}G$ such that $SF(\pi', G) = 0$.

Proof. Setting $d' = T - d$, i.e., until the end of the game, ensures that no further infostate can induce strategy fusion, as there are no infostates remaining. \square

V. RESULTS

A. Games

Our experiment set involved testing five games: Card Game, Battleship, Dark Chess, Phantom Tic-Tac-Toe, and Dark Hex. Each of them is considered a large game, is described below and is implemented in OpenSpiel [25], a collection of environments and algorithms for research in general reinforcement learning and search/planning in games. The benchmarks were chosen to show the strengths and weaknesses of the algorithm, especially, by choosing games with public observations (Card game, Battleship) or with private observations (Phantom Tic-Tac-Toe, Dark Chess, and Dark Hex).

Public observations refer to information that is visible to all players, while private observations are exclusive to a player. This distinction significantly influences the dynamics of strategy fusion. Private observations amplify the number of potential world states within a single infostate, thereby

increasing the likelihood of strategy fusion. As our method reduces strategy fusion, we expect EPIMC to be more effective in games where observations are private.

a) Card game: The game is played with two players, 22 cards are taken from a pack of 52, and known by all, 6 are hidden and 8 is given to each player. The playing phase is decomposed into tricks, the player starting the trick is the one who won the previous trick. The starting player of a trick can play any card in his hand, but the other player must follow the suit of the first player. If he can not, he can play any card he wants without possibly winning the trick. The winner of the trick is the one with the highest-ranking card. At the end of the game, a player wins if he has at least half of the number of tricks won.

b) Battleship: Battleship is a two-player strategy-type guessing game. Each player possesses a grid. In the beginning, each player secretly places a set of ships S on their grid. After placement, turn after turn, each player tries to fire at other players' ships. The game ends when all the ships of a player have been destroyed. The payoff of each player is computed as the sum of the opponent's ships that were destroyed, minus the sum of ships that the player lost. The grid is fixed to 3×3 , with 2 ships, one of size 1×1 , and the second of size 2×1 .

c) Dark Chess: Dark chess is a chess variant with incomplete information. In chess, there are two players, white and black, each controlling a set of chess pieces of their respective colors. The goal of the game is to checkmate the opponent's king. In Dark chess, the incomplete information comes from the fact that each player sees his own pieces, but only sees his opponent's pieces if they are reachable by one of his pieces. Furthermore, in his variant, the purpose is to capture the king (not to checkmate it), however, a player must be wary as he is not told if their king is in check. The size of the board is fixed to 4×4 .

d) Phantom Tic-Tac-Toe: Phantom Tic-Tac-Toe is a variant of the game of Tic-Tac-Toe with imperfect information. In Tic-Tac-Toe, the goal is to claim three cells along the same row, column, or diagonal. With imperfect information, the players do not observe the other player's pieces, only a referee knows the world state of the board. When it is a player's turn, the player selects a move and indicates it to the referee. The referee informs the player's whether the action is 'legal' or 'illegal'. If the move is 'illegal', the player must choose a new move until they find a legal one.

e) Dark Hex: Dark Hex is an imperfect information version of the classic game of Hex. The objective of the game is to create a connection between opposite sides of a rhombus-shaped board. In Dark Hex, players are not exposed to opposite sides' pieces of information. Only a referee has the full information of the board and when a move fails due to collision/rejection the player gets some information of the cell and is allowed to make another move until success. The size of the board is fixed to 4×4 .

B. Experimental Information

For all the experiments, the budget ranged from 0.1 seconds to 100 seconds for Card Game, Battleship and Phantom Tic-Tac-Toe, and from 0.1 seconds to 1000 seconds for Dark Hex and Dark Chess. Each experiment was conducted over 500 games, in which the games were evenly split between playing in the first and second positions. The opponent is PIMC with a fixed one-second budget.

All experiments were executed on a single CPU Intel(R) Xeon(R) Gold 5218. EPIMC and PIMC use random rollout as the perfect information leaf evaluator, depth at 3 and Information Set Search for the subgame resolution. In practice, PIMC is often tested with minimax as the perfect information leaf evaluator, yet using it may be slow in large benchmarks or require using a handmade heuristic. As a reminder, EPIMC at depth 1 is PIMC. To reduce EPIMC/PIMC costs, multiple CPUs could be utilized, but for fairness across algorithms, this approach was not employed.

Other online algorithms: The other algorithms compared are Information Set MCTS (IS-MCTS) [11], Online Outcome Sampling (OOS) [26], Recursive PIMC (IIMC) [3], and a random agent (Random). IS-MCTS is a determinization-based algorithm that employs Monte Carlo Tree Search on infostates. OOS is a regret-based algorithm that converges to the Nash equilibrium with increasing search time. IIMC is a determinization-based algorithm rooted in PIMC, estimating action values by recursively calling PIMC until game completion.

For IS-MCTS, the exploration constant was chosen from 0.6, 1, 1.5, 2 and set at 1. In OOS, the target was selected between Information Set and Public Subgame Targeting, and it was set at Information Set. Regarding IIMC, the number of samplings at level 2 was chosen from 2, 5, 10 and set at 5.

C. Experimental Results

In our experiments, we aimed (i) to analyze the hyperparameters of EPIMC (depth, subgame resolution, perfect leaf evaluator) and (ii) to compare its performance against other online algorithms.

a) Postponing leaf evaluator: In Figure 2, we examine the performance of EPIMC according to various depths varying from 1 to 3. As expected, for Card Game and Battleship, increasing the depth does not lead to any improvement as both games have a majority of their observation public. Conversely, for games where the observations are private, we observe an important increase in performance. At 100 seconds for Dark Chess, we achieve 80%/65%/45% winning rates at depths 3/2/1. More than that, at 1000 seconds, EPIMC at depth 3 wins close to 100% for both Dark Hex and Dark Chess. Interestingly, in Dark Hex, at 1000 seconds, similar performances are observed at depth 2 and 1. Indeed reducing the strategy fusion does not necessarily mean that the strategy produced at the end will be changed.

In the following, the experiments are reduced to Dark Chess, Phantom Tic-Tac-Toe, and Dark Hex.

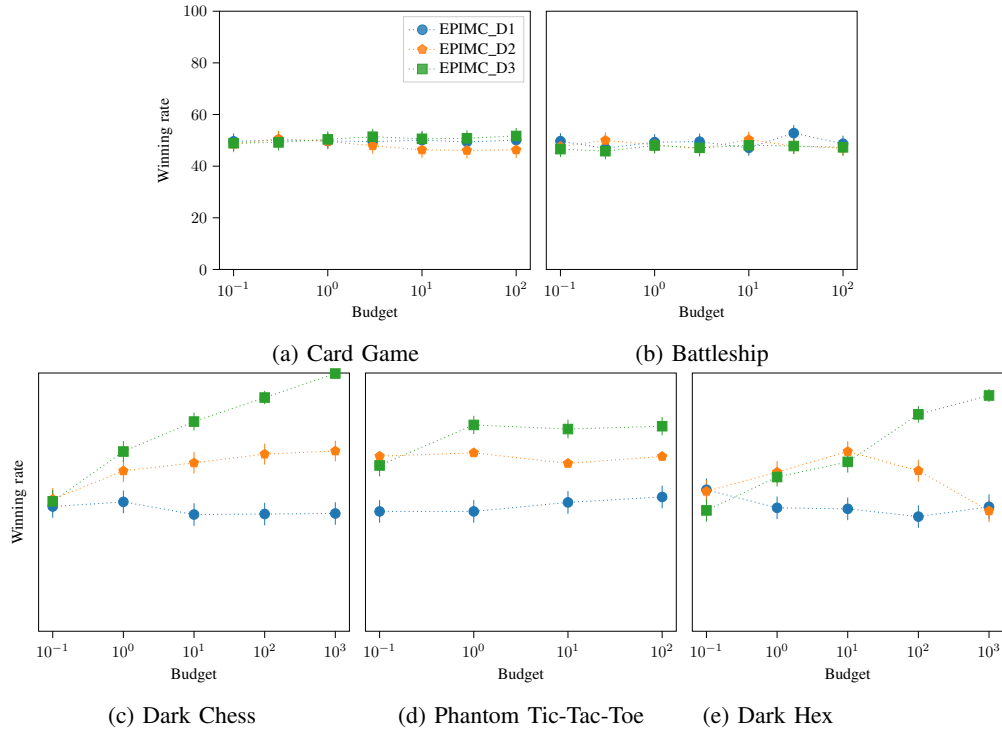


Fig. 2: Winning rate of EPIMC when the depth from 1 to 3. The opponent is PIMC with one second of the budget.

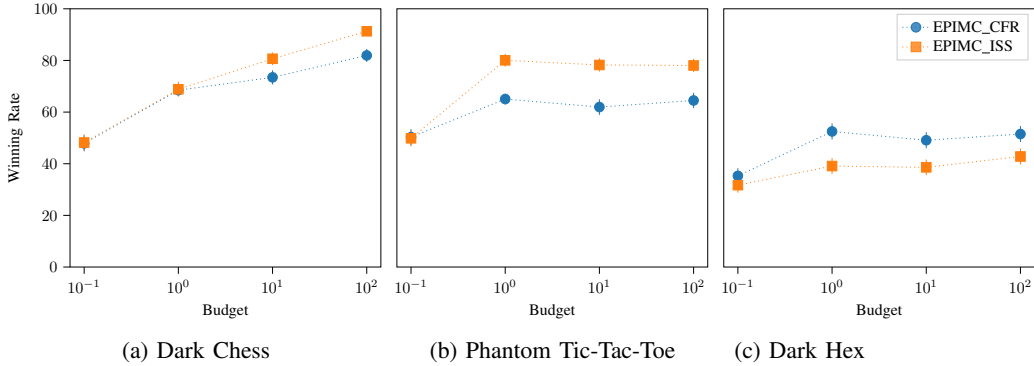


Fig. 3: Winning rate of EPIMC when the subgame is CFR+ or ISS. The opponent is PIMC with one second of the budget.

b) Subgame resolution: In Figure 3, we compare CFR+ against Information Set Search (ISS) in the Extended game resolution. CFR+ is used with 1000 iterations. Using CFR+ in Dark Hex 4×4 was too costly in computational time, and using it on Dark Hex 3×3 . As can be observed, neither method is superior to the other, worse performances are obtained with CFR+ in Dark Chess and Phantom Tic-Tac-Toe but better performance is achieved in Dark Hex.

c) Perfect information leaf evaluator: In Figure 4, we compare Minimax with alpha-beta pruning against random rollout for the perfect information leaf evaluator. Due to the important cost of using Minimax, the test was not conducted in Dark Chess, and as before, the size Dark Hex was reduced to 3×3 . As before, neither method is superior to the other,

as worse performance is obtained with Minimax in Phantom Tic-Tac-Toe but stronger performance is achieved in Dark Hex. However, the differences between the two methods are important in Dark Hex, where Minimax obtains performance close to 70% while Random Rollout obtains performance close to 50%.

d) Against other online algorithms: In Figure 5, a comparative analysis between EPIMC and various online algorithms against opponents is presented. It is evident from the results that EPIMC and IS-MCTS exhibit superior performance compared to other algorithms. Notably, EPIMC achieved remarkable success, particularly at depth 3, outperforming IS-MCTS across all benchmarks considered. Furthermore, it is noteworthy that even at a depth of 2, EPIMC demonstrated

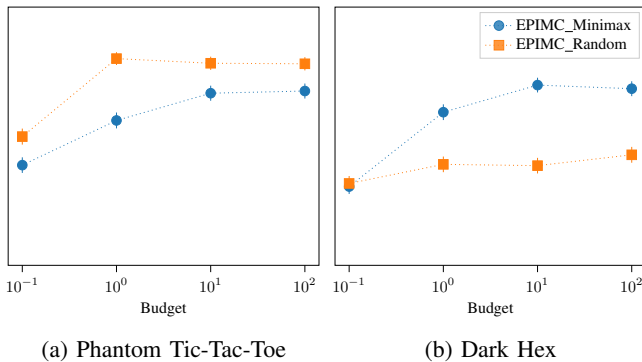


Fig. 4: Winning rate of EPIMC when the perfect information leaf evaluator is Minimax or Random Rollout. The opponent is PIMC with one second of the budget.

comparable or even superior performance to IS-MCTS in games like Dark Chess and Phantom Tic-Tac-Toe.

D. Discussion

As we have seen and as was expected, increasing the depth has a significant impact on games with a private observation (Dark Hex / Dark Chess / Phantom Tic-Tac-Toe) and much less impact on games without much private information (Card Game / Battleship). On the domains tested and given a sufficient budget, augmenting the depth consistently outperforms the basic version. In addition, a depth of 2 is sufficient, in most cases, to beat the state-of-the-art online algorithms. With regard to the other hyperparameters, we recommend using Information Set Search and Random Rollout as they require less computing time and therefore can be computed on larger games. However, customization could be beneficial in order to increase performance or the need to maintain theoretical properties in the subgame. Especially, using algorithms such as CFR+ allows to obtain the properties of CFR+ (in the subgame) such as the convergence towards the Nash equilibrium in two players.

VI. RELATED WORK

In determinization-based algorithms, both IIMC [3] and IS-MCTS [11] extend reasoning beyond a depth of one. IS-MCTS increases depth with the budget, and IIMC recursively calls PIMC until the game’s end. In both cases, as the depth exceeds PIMC, it is expected that strategy fusion will be reduced. However, there is no guarantee that it can be entirely eliminated, and it is challenging to quantify the extent of its reduction. This is because the two methods do not uniformly explore the state space, and worse, they use exploration/exploitation mechanisms based on estimates that are distorted by the presence of strategy fusion.

PIMC and our work share similarities with Unsafe/Safe Subgame Solving [27]. However, unlike Unsafe/Safe algorithms, our method does not require an expensive pre-computed value function. Unsafe/Safe algorithms often solve

an abstraction of the game that includes the subgame with leaf estimation and refine this during play. Moreover, their methods rarely detail subgame construction during the game, unlike our determinization method, which uses sampling and fast leaf evaluators.

In a study by Zhang et al. [28], a similar approach used a perfect leaf evaluator at an extended depth compared to PIMC, focusing on ‘common knowledge’ to reduce costs without abstraction. Their research did not explore determination algorithms or key aspects central to our investigation, such as sub-game creation, budget considerations, and the influence of depth on strategy fusion and private/public observations. Additionally, their move ordering, prioritizing promising nodes over uniform state space exploration, may lead to strategy fusion and suboptimal decision-making, issues we address in our study.

VII. CONCLUSION AND FUTURE WORKS

In this paper, we introduce a novel online algorithm ‘Extended PIMC’ for games with imperfect information. Building upon the foundation of PIMC, our approach postpones the perfect leaf evaluator to a deeper depth. Thanks to that, we have been able to successfully reduce past problems of PIMC and beat other online algorithms on multiple benchmarks. Especially, when benchmarks have hidden observation, significant performance improvements are observed. Furthermore, we conducted an in-depth analysis of various hyperparameters to provide a comprehensive understanding of their impact.

We enhance our research by presenting theoretical foundations for determinization-based algorithms that suffer from strategy fusion. We demonstrate that, in the worst case, increasing the depth does not increase the strategy fusion and in every case, there exists a depth d such that the strategy fusion is strictly reduced.

As our algorithm is online, it has the advantage of being tested in a short period of time, especially in comparison to recent algorithms which need a domain-specific abstraction or a very high initialization cost due to neural networks. Even though, improving our algorithm by using deep learning could have led to superior performance. In particular, this could provide a better and faster approximation to the leaf or being able to remove the problem of non-locality by adding an inference system [29].

In future research, it would be intriguing to assess our algorithm in conjunction with other determination algorithms. Particularly, exploring a trade-off between our approach, which ensures non-strategy fusion at a certain depth thanks to uniform sampling, and other algorithms that may explore the state space more efficiently but currently encounter issues with strategy fusion during exploration, could be beneficial. Such an investigation could shed light on the strengths and limitations of different approaches and potentially lead to the development of more robust and efficient algorithms for imperfect information games.

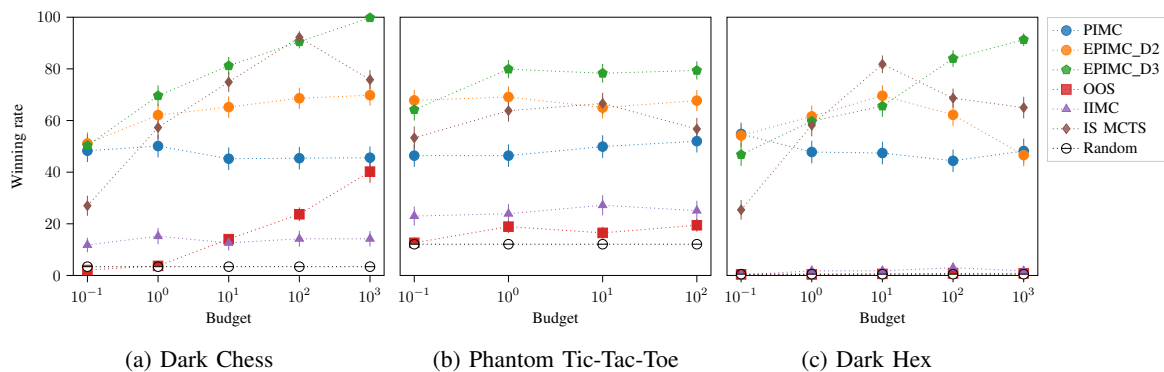


Fig. 5: Winning rate of online algorithms. The opponent is PIMC with one second of the budget.

REFERENCES

- [1] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, pp. 484–489, 2016.
- [2] N. Brown and T. Sandholm, "Superhuman AI for multiplayer poker," *Science*, vol. 365, pp. 885 – 890, 2019.
- [3] T. Furtak and M. Buro, "Recursive Monte Carlo search for imperfect information games," *2013 IEEE Conference on Computational Intelligence in Games (CIG)*, pp. 1–8, 2013.
- [4] B. Bouzy, A. Rimbaud, and V. Ventos, "Recursive Monte Carlo Search for Bridge Card Play," *2020 IEEE Conference on Games (CoG)*, pp. 229–236, 2020.
- [5] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. P. Lillicrap, K. Simonyan, and D. Hassabis, "Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm," *ArXiv*, vol. abs/1712.01815, 2017.
- [6] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel *et al.*, "A general reinforcement learning algorithm that masters chess, shogi, and go through self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018.
- [7] T. W. Neller and M. Lanctot, "An introduction to counterfactual regret minimization," in *Proceedings of model AI assignments, the fourth symposium on educational advances in artificial intelligence (EAAI-2013)*, vol. 11, 2013.
- [8] O. Tammelin, N. Burch, M. B. Johanson, and M. Bowling, "Solving Heads-Up Limit Texas Hold'em," in *IJCAI*, 2015.
- [9] M. Lanctot, K. Waugh, M. A. Zinkevich, and M. Bowling, "Monte Carlo Sampling for Regret Minimization in Extensive Games," in *NIPS*, 2009.
- [10] J. R. Long, N. R. Sturtevant, M. Buro, and T. Furtak, "Understanding the Success of Perfect Information Monte Carlo Sampling in Game Tree Search," in *AAAI*, 2010.
- [11] P. I. Cowling, E. J. Powley, and D. Whitehouse, "Information Set Monte Carlo Tree Search," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, pp. 120–143, 2012.
- [12] T. Cazenave and V. Ventos, "The $\alpha\mu$ Search Algorithm for the Game of Bridge," in *Monte Carlo Search at IJCAI*, ser. Communications in Computer and Information Science, 2021.
- [13] M. Moravčík, M. Schmid, N. Burch, V. Lisý, D. Morrill, N. Bard, T. Davis, K. Waugh, M. B. Johanson, and M. H. Bowling, "DeepStack: Expert-level artificial intelligence in heads-up no-limit poker," *Science*, vol. 356, pp. 508 – 513, 2017.
- [14] Q. Jiang, K. Li, B. Du, H. Chen, and H. Fang, "DeltaDou: Expert-level Doudizhu AI through Self-play," in *IJCAI*, 2019.
- [15] Y. Zhang, D. Yan, B. Shi, H. Fu, Q. Fu, H. Su, J. Zhu, and N. Chen, "Combining Tree Search and Action Prediction for State-of-the-Art Performance in DouDiZhu," in *IJCAI*, 2021.
- [16] M. Schmid, M. Moravčík, N. Burch, R. Kadlec, J. Davidson, K. Waugh, N. Bard, F. Timbers, M. Lanctot, Z. Holland, E. Davoodi, A. Christian-son, and M. H. Bowling, "Player of Games," *ArXiv*, vol. abs/2112.03178, 2021.
- [17] I. Frank and D. A. Basin, "Search in Games with Incomplete Information: A Case Study Using Bridge Card Play," *Artif. Intell.*, vol. 100, pp. 87–123, 1998.
- [18] M. J. Schofield and M. Thielscher, "General Game Playing with Imperfect Information," *J. Artif. Intell. Res.*, vol. 66, pp. 901–935, 2019.
- [19] V. Kovařík, M. Schmid, N. Burch, M. H. Bowling, and V. Lisý, "Rethinking Formal Models of Partially Observable Multiagent Decision Making," *Artif. Intell.*, vol. 303, p. 103645, 2022.
- [20] D. E. Knuth and R. W. Moore, "An analysis of alpha-beta pruning," *Artificial Intelligence*, vol. 6, no. 4, pp. 293–326, 1975. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/0004370275900193>
- [21] C. Browne, E. J. Powley, D. Whitehouse, S. M. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. P. Liebana, S. Samothrakis, and S. Colton, "A Survey of Monte Carlo Tree Search Methods," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, pp. 1–43, 2012.
- [22] J. Arjoniella, A. Saffidine, and T. Cazenave, "Mixture of public and private distributions in imperfect information games," in *2023 IEEE Conference on Games (CoG)*. IEEE, 2023, pp. 1–8.
- [23] A. Parker, D. Nau, and V. Subrahmanian, "Overconfidence or paranoia? search in imperfect-information games," in *PROCEEDINGS OF THE NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE*, vol. 21. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006, p. 1045.
- [24] —, "Paranoia versus overconfidence in imperfect information games," in *AAAI Proceedings of the 21st National Conference on Artificial Intelligence*, vol. 2, 2006.
- [25] M. Lanctot, E. Lockhart, J.-B. Lespiau, V. F. Zambaldi, S. Upadhyay, J. Pérolat, S. Srinivasan, F. Timbers, K. Tuyls, S. Omidshafiei, D. Hennes, D. Morrill, P. Muller, T. Ewalds, R. Faulkner, J. Kramár, B. D. Vyllder, B. Saeta, J. Bradbury, D. Ding, S. Borgeaud, M. Lai, J. Schrittwieser, T. W. Anthony, E. Hughes, I. Danihelka, and J. Ryan-Davis, "OpenSpiel: A Framework for Reinforcement Learning in Games," *ArXiv*, vol. abs/1908.09453, 2019.
- [26] V. Lisý, M. Lanctot, and M. Bowling, "Online Monte Carlo Counterfactual Regret Minimization for Search in Imperfect Information Games," in *AAMAS*, 2015.
- [27] N. Burch, M. B. Johanson, and M. Bowling, "Solving Imperfect Information Games Using Decomposition," in *AAAI*, 2014.
- [28] B. Zhang and T. Sandholm, "Subgame solving without common knowledge," *Advances in Neural Information Processing Systems*, vol. 34, pp. 23 993–24 004, 2021.
- [29] D. Rebstock, C. Solinas, M. Buro, and N. R. Sturtevant, "Policy Based Inference in Trick-Taking Card Games," *2019 IEEE Conference on Games (CoG)*, pp. 1–8, 2019.