The Machine Reconnaissance Blind Chess Tournament of NeurIPS 2022

Ryan W. Gardner RYAN.GARDNER@JHUAPL.EDU
Gino Perrotta GINO.PERROTTA@JHUAPL.EDU
Johns Hopkins University Applied Physics Laboratory
Anvay Shah SHAH.ANVAY@GMAIL.COM
Shivaram Kalyanakrishnan SHIVARAM@CSE.IITB.AC.IN
Indian Institute of Technology Bombay
Kevin A. Wang KEVINWANG@KEVINWANG.US
Meta AI
Gregory Clark GREGORYCLARK@GOOGLE.COM
ML Collective
Google
Timo Bertram TBERTRAM@FAW.JKU.AT
Johannes Fürnkranz JUFFI@FAW.JKU.AT
Johannes Kepler Universität Linz
Martin Müller MMUELLER@UALBERTA.CA
University of Alberta
Brady P. Garrison BRADY_GARRISON@BROWN.EDU
Brown University
Prithviraj Dasgupta RAJ.DASGUPTA@NRL.NAVY.MIL
U.S. Naval Research Laboratory
Saeid Rezaei SAEID.REZAEI@CS.UCC.IE
University College Cork

Editors: Marco Ciccone, Gustavo Stolovitzky, Jacob Albrecht

Abstract

Reconnaissance Blind Chess is a game that plays like regular chess but rather than continuously observing the entire board, each player can only momentarily and privately observe selected board regions. It has imperfect information and little common knowledge. The Johns Hopkins University Applied Physics Laboratory (the game's creator) and several partners organized the third NeurIPS machine Reconnaissance Blind Chess competition in 2022 to bring people together to attempt to tackle research challenges presented by the game. 18 bots played each other in 9,180 games (60 matches per bot pair) over 4 days. The top bot exceeded the performance of all of last year's bots yet a practical, sound (unexploitable) algorithm remains unknown.

Keywords: reconnaissance blind chess, imperfect information, reinforcement learning, common knowledge

1. Introduction

Games are well-defined abstractions for general decision-making processes encapsulating sequential observations, choices, and effects of those choices as agents interact with their environment and each other. Games have been central to many famous breakthroughs in artificial intelligence (AI) including the development of superhuman systems Deep Blue (Campbell et al., 2002), AlphaGo (Silver et al., 2016, 2017), AlphaZero (Silver et al., 2018), Libratus (Brown and Sandholm, 2017a), DeepStack (Moravčík et al., 2017), and Pluribus (Brown and Sandholm, 2019).

Games with large amounts of private knowledge and particularly little common knowledge and strategic planning have not been studied as extensively as others, however. Even a round of Texas Hold'em poker, for example, a game that is famous for including imperfect information, only has 2 hidden cards per player and lacks interesting strategy in the absence of the uncertainty. Stratego is a recent partial exception (Perolat et al., 2022) in the sense that it has large amounts of private knowledge and requires strategic planning. However, the game has significant common knowledge, an algorithm for effective search in Stratego remains an open question, and superhuman play has yet to be achieved. Reconnaissance Blind Chess (RBC) (Newman et al., 2016) was created as a challenge for strategic decision-making and planning in the face of uncertainty. Algorithms that were used to develop famous poker systems break down in RBC due to their dependence on public information (or common knowledge) while algorithms for games like regular chess break down due to dependence on perfect information. In competitive non-game (real-world) settings, parties are not forced or incentivized to make information public. Rather, they want to keep as much information private as possible, so making strategic decisions with large amounts of uncertainty is common.

For these reasons, the Johns Hopkins University Applied Physics Laboratory (JHU/APL) and several partners have been hosting machine RBC competitions as part of NeurIPS (Gardner et al., 2020; Perrotta et al., 2022). Several of the competition organizers and bot authors wrote this paper to summarize the research challenges, algorithms, and results from the competition held in 2022. Although the best bot in 2022 is an improvement from previous competitions, none of the algorithms used converges to an optimal strategy or has directly addressed all the game's research challenges. RBC continues to be a challenging problem for AI research.

2. Rules of Reconnaissance Blind Chess

RBC plays like chess with a few key differences. The pieces start in the same places as in chess on a standard 8×8 chessboard and move in the same way. However, (as a blind chess variant) players cannot directly observe their opponents' pieces. Instead, prior to making each move, a player chooses a 3×3 region of the board to *sense*; that player is immediately informed of any pieces and their associated locations in that region at that time. The opponent is not informed of the area sensed. For this reason, the game must be facilitated by a third party (a computer in modern practice).

Both players are informed of all captures and the square where they occurred. Neither player is informed of the captured or capturing piece.

Several other modifications are made to accommodate the imperfect information. There is no concept of check. The game is won by capturing the opponent's king or if the opponent runs out of time. Players may attempt to make moves that would be illegal in standard chess because they are unaware of opponent pieces in the way. When a move is attempted, pieces make the longest move that would be legal along the path of the attempted move (again with the notion of check removed), including captures, and then stop. For example, a rook attempting to move through an opponent piece, would capture that opponent piece and stop. The moving player is informed of their piece's stopping location.

We ran this tournament with a time limit of 15 minutes per game plus 5 seconds per turn (sense and move), per player. A draw is called (automatically) if 50 consecutive moves have no captures or pawn moves.

The private sense action is fairly unique to RBC, and is the root of the core research questions by making a agent's key observations unknown to other agents.

Interested readers can get the complete list of rules and play at https://rbc.jhuapl.edu. We also make all past games available for download.

3. Research Challenges and Related Work

The core algorithms used to develop playing strategies for games of perfect information are typically minimax and Monte Carlo tree search (MCTS) (Kocsis and Szepesvári, 2006; Browne et al., 2012). These search algorithms are usually limited in depth by using an estimate of the value of a reached state. AlphaZero (Silver et al., 2017, 2018), for example, used deep reinforcement learning to create a neural network that estimated the value of states as well as move probabilities to help guide MCTS. However, these algorithms do not yield optimal strategies, in general, in games of imperfect information. While the optimal strategy in a perfect information game is deterministic, in general, with imperfect information, it is not (else a player would be completely predictable). Additionally, with perfect information, the past is irrelevant, but with imperfect information, even currently unreachable portions of a game may impact the optimal strategy from the current state.

Core algorithms for converging on optimal (Nash equilibrium) strategies in two-player zero-sum games with imperfect information include fictitious self-play (FSP) (Heinrich et al., 2015) and counterfactual regret minimization (CFR) (Zinkevich et al., 2008). In their unmodified forms these algorithms are intended to compute a policy offline (prior to game-play); they iteratively sample from the start of the game. Simplifying this process to estimate the strategy from a point many moves into the game is non-trivial because past, unreachable portions of the game potentially affect probabilities of current states. RBC is too large to sample from the start and sufficiently sample states several moves into the game.

Approaches have been developed to limit both the breadth and depth of samples taken using CFR online (during game-play) while continuing to approximate Nash equilibria. The primary approach for limiting breadth decomposes the game along public belief states, where a public belief state includes the closure of all possible game histories that are indistinguishable by at least one agent, i.e., histories where the public knowledge is the same (Burch et al., 2014; Moravčík et al., 2017; Brown and Sandholm, 2017b; Sustr et al., 2019; Brown et al., 2020). These approaches were central to the creation of the first superhuman poker

bots (Brown and Sandholm, 2017a; Moravčík et al., 2017). Because RBC has little public information, these approaches do not apply well to RBC. (Each public state would include a large portion of the game.) Online outcome sampling (OOS) (Lisý et al., 2015) is an approach that limits breadth in a sense by focusing samples on a breadth-restricted portion of the game (e.g., the portion that is consistent with the current game's history) via importance sampling. It is currently unable to practically produce a strong RBC strategy partly because all possible opponent senses are consistent with the current game's history; the number of possible sense sequences increases exponentially with the number of turns taken.

Other approaches have limited search depth by using neural networks to approximate values of all possible player states within each possible public state (Moravčík et al., 2017; Brown et al., 2020). Again, because of the large amount of private information in RBC, a public state frequently includes a large portion of the game. The number of possible player states within a public state can again grow exponentially making this approach impractical in its current form. Another approach (Brown et al., 2018) creates multiple strategies an opponent could use after the depth limit to approximate the value of a state. This approach does not rely on public information but it does require approximating a Nash equilibrium policy for the entire game and either computing an opponent's best response to strategies developed each turn or modifying the Nash equilibrium policy in ways that would, in culmination, meaningfully account for possible errors in the policy. Practical means of doing these things have not been demonstrated for a game as complex as RBC.

Knowledge-limited subgame solving (KLSS) (Zhang and Sandholm, 2021) is an approach created to compute strategies online when there is little common knowledge (public information). They limit breadth by *knowledge distance*, where they define 0 to be the distance to states consistent with the acting player's current knowledge, and a state's distance is $n + 1$ if it is possible from a player's perspective from one of the other player's distance n states. Even the number of states at distance 0 may grow exponentially in RBC (again, due to the private sense), so raw KLSS limited to distance 0 seems impractical for RBC. The authors further discuss reducing states that are indistinguishable after a certain point in time, but this could disregard the order of sense actions in RBC, which may be significant.

Approaches have also been created for approximating optimal strategies for imperfect information games offline including neural fictitious self play (NFSP) (Heinrich and Silver, 2016), deep counterfactual regret minimization (Deep-CFR) (Brown et al., 2019), DREAM (Steinberger et al., 2020), ESCHER (McAleer et al., 2022), and regularized Nash dynamics (R-NaD) (Perolat et al., 2022). By using neural networks, these approaches can generalize across fairly large state-spaces, demonstrating expert-level play in Stratego (Perolat et al., 2022), and have been shown empirically to converge to a Nash equilibrium. However, online search has been shown to greatly improve play strength (Silver et al., 2017), these methods have not achieved superhuman play in complex games without search, and it is not clear that these approaches are practical on large games like RBC without extraordinary hardware.

Other common blind chess variants include Kriegspiel (Ciancarini and Favini, 2010; Russell and Wolfe, 2005) and Dark Chess (Zhang and Sandholm, 2021). In Kriegspiel, players only observe the board through attempted moves. In Dark Chess, players can observe the squares to which their pieces may legally move. Stratego is similar to RBC in

that it involves a large amount of private information and strategic planning. In Kriegspiel, Dark Chess, and Stratego, observations are tied to piece placement. We estimate that this reduces the amount of private information compared to RBC although the games present similar research challenges.

4. Competition Structure

The competition consisted of a tournament that started on October 19, 2022 (although we also hosted several optional test tournaments). Anyone was welcome to compete at no cost. Bot authors ran their own code and bots participated in the tournament by communicating with our server (potentially using our bot-development kit or using their own communication code).

In the tournament, each bot played each other bot 60 times, 30 as white and 30 as black. We used the Bayesian Elo rating (Coulom, 2006) of each bot to determine rank. Game order does not affect Bayesian Elo computation.

Bots were given 15 minutes per game plus 5 seconds per turn total time to select their sense and move actions, and participants were required to support playing 4 simultaneous games. The tournament last about 70 hours.

Affiliates of the Johns Hopkins University and organizers themselves were welcome to submit bots but were not eligible for prizes.

5. Overview of Approaches

Our NeurIPS 2022 competition included a variety of approaches. We briefly summarize the algorithms for several of the bots in Table 1 in descending rank. Table 2 further summarizes high-level features of each bot.

Table 1: Brief description of select bots' algorithms.

Bot Name	**Approach**
StrangeFish2	Further development of the bot of the same name which competed in NeurIPS 2021 (Perrotta et al., 2022). It tracks all possible locations of opponent pieces and estimates position strengths and likelihoods for each. Both sense and move decisions are made to maximize the expected outcome of the current turn. Improved performance was driven by developments in board state value heuristics – which continue to augment Stockfish engine analysis – and by prioritized sub-sampling of tracked positions when time limits did not permit evaluation of all hypotheses.
Fianchetto	Makes minor changes to last year's version of Fianchetto, which is based on StrangeFish (Perrotta and Perrotta, 2019) and uses Leela Chess Zero (Lc0) to model a probability distribution over opponent board states as a POMDP. The most notable change is an increased weight to defensive sensing and moving to detect opponent attacking moves that have been given a low probability. These weights are opponent-specific, prioritizing more defensive play against more aggressive opponents. Additionally, chooses its first move from an opening book, designed to give preference to closed chess positions, with the rationale that closed positions tend to have lower uncertainty and smaller information set sizes.

Bot Name	Approach
Kevin	Explicitly keeps track of a distribution over possible game histories. (Opponent senses matter, not only board state.) Predicts opponent actions (both senses and moves) using a neural network. The neural network is trained with imitation learning on game logs of StrangeFish2. The neural net inputs come from OpenSpiel and its RBC implementation, and they pass through a residual neural network (ResNet) and a long short-term memory (LSTM) network, which output a distribution over legal actions. Selects the best move by maximizing expected value over the distribution of histories, using Stockfish to evaluate board states.
Châteaux	Based on deep synoptic Monte Carlo planning (Clark, 2021). Approximates information states with a lossy stochastic abstraction, encoding sets of boards with compact fixed-size synopses. The fixed-size representations are used as input to a residual neural network which outputs a value estimation and a policy distribution over actions. The network was trained on approximately 600,000 games available online. The bot maintains a "second-order" belief state of the opponent's information with an unweighted particle filter made up of possible approximate information states from the opponent's perspective. The bot plans with playouts according to a variant of Monte Carlo tree search guided by the neural network. A list of all possible piece placements is used to eliminate impossible states in the particle filter and to initialize playouts for the tree search algorithm.
ROOKie	Comprehensive multi-hypothesis-tracking of piece positions on the board, with actions selected by voting among tracked boards. Each tracked board contributes votes of equal weight for up to 5 moves, against any moves that leave the king in check, and is considered neutral on all remaining moves. Computing top moves is often *much* faster than analysis of position strength, enabling the bot to evaluate more boards than if it were analyzing position. Sense actions are taken to maximize expected consensus in votes from remaining boards.
Oracle	Exhaustively tracks board states. Chooses sense action to identify possible checks, or otherwise minimizes the expected number of possible board states. Chooses the move that is recommended by Stockfish most across all possible board states.
Marmot	Uses a Monte Carlo counterfactual regret minimization (MC-CFR) algorithm (Lisý et al., 2015) for sensing and moving with novel modifications to make assumptions about the past and start search from mid-game (sacrificing soundness). It uses a heuristic evaluation function, which includes Stockfish's board-evaluation, based on actual board state and a tracked uncertainty measurement to evaluate the intermediate states reached from action sequences sampled using MC-CFR. Tracks all possible opponent board states for the current time and past timesteps based on current observations and uses subsets of those for starting its MC-CFR searches.
JKU-CODA	Uses contextual preference ranking (Bertram et al., 2021) to compute a probability distribution over the possible board states based on historical RBC games. Evaluates all possible moves on the most likely boards with Stockfish and chooses the move with the best combination of worst-case and weighted-average evaluation. Senses to maximize the reduction in the number of possible board states and to decrease the uncertainty in the evaluations of the most promising move. The contextual preference ranking uses a Siamese neural network to generate an embedding of boards and observations, which is transformed into the probability distribution used for weighting.
DynamicEntropy	A pessimistic variant of single observer information set Monte Carlo tree search (Cowling et al., 2012). This considers the game only from the searching player's perspective, permitting the simulated opponent to select observations directly instead of their own sense and move actions. Leaf nodes are scored by shallow Stockfish evaluation of randomly sampled possible board states.
trout (baseline)	Maintains a single board-state estimate that is formed directly from the latest observation of each square. Chooses the move recommended by Stockfish for its board estimate. If a piece was just captured or it thinks it will capture a piece next turn, it senses over the capture square. Otherwise it chooses a random location to sense that does not contain any of its own pieces.

Bot Name	**Approach**
attacker (baseline)	Randomly chooses and executes one of four scripted attack sequences. If that fails, no more moves are made.
GarrisonNRL	Maintains a set of possible boards. Selects sensing action as the location that maximizes a sensing weight equal to the product of the number of potential opponent moves to a location and the value of pieces moving to those locations. When the board set size is below 50, utilizes chess engines and heuristic approaches to select its move. For each board in the board sample, chooses a best move by attacking the opponent king if possible, otherwise by using Lc0 with the Maia 1700 network (McIlroy-Young et al., 2020). Selects a movement action by using the most common best move. With larger board sets, utilizes a single observer information set Monte Carlo tree search approach (Cowling et al., 2012).
uccchess	Based on StrangeFish (Perrotta and Perrotta, 2019), but has a set of 20 neural networks to replace the chess engine. It evaluates the game history each turn and matches the opponent to one of 20 different classes, which correspond with the 20 neural networks. Using the network for the matched class as a replacement for the chess engine in StrangeFish, it computes a score for each action. The opponent's class is selected by an agent that is trained using reinforcement learning. During its training, uccchess plays againts several predefined bots, and the results of the matches provide a reward signal.
random (baseline)	Chooses moves uniformly at random. (Senses are irrelevant.)

Table 2: High-level comparison of features included in competing algorithms.

Bot	Elo	Rank	Tracks all board states	Tracks opponent infosets	Tracks all game states	Models opponent moves	Models opponent senses	Uses opponent-specific models	Uses a chess engine	Senses to minimize states	Uses a custom neural network
StrangeFish2	1762	1	•						•		
Fianchetto	1644	2	•			•		•	•	•	
Kevin	1623	3	•	•		•	•		•		•
Châteaux	1621	4	•	•		•	•	•			•
ROOKie	1551	5	•						•		
Oracle	1465	6	•						•	•	
Marmot	1329	7	•	•		•	•		•		
JKU-CODA	1283	8	•						•		•
DynamicEntropy	1194	9	•						•		
trout	1116	11							•		
attacker	1099	12									
GarrisonNRL	1039	13	•			•			•	•	•
uccchess	1025	14	•					•			•
random	893	15									

6. Results and Observations

Figure 1 presents a crosstable of each bots' wins, losses, and draws against each other bot. StrangeFish2 won definitively with 68 total wins more than the bot with the next greatest number and a winning record against every other bot. Second-place was won less decisively:

	Overall	StrangeFish2	Fianchetto	Kevin	Châteaux	ROOKie	Oracle	Marmot	JKU-CODA	DynamicEntropy	SomeRegret	trout	attacker	GarrisonNRL	uccchess	random	arandombot	srcork	uccch
StrangeFish2	905-105-10		34-21-5	31-29	45-14-1	51-7-2	46-13-1	55-5	57-3	59-1	56-4	59-1	60-0	60-0	52-7-1	60-0	60-0	60-0	60-0
Fianchetto	837-168-15	21-34-5		31-29	18-42	37-18-5	49-9-2	52-8	51-9	58-2	55-5	60-0	59-1	57-3	49-8-3	60-0	60-0	60-0	60-0
Kevin	825-195	29-31	29-31		32-28	40-20	43-17	45-15	57-3	56-4	46-14	56-4	54-6	53-7	48-12	58-2	60-0	60-0	59-1
Châteaux	825-193-2	14-45-1	42-18	28-32		31-28-1	40-20	51-9	56-4	51-9	58-2	60-0	60-0	57-3	37-23	60-0	60-0	60-0	60-0
ROOKie	776-236-8	7-51-2	18-37-5	20-40	28-31-1		35-25	51-9	47-13	60-0	52-8	60-0	46-14	59-1	53-7	60-0	60-0	60-0	60-0
Oracle	713-303-4	13-46-1	9-49-2	17-43	20-40	25-35		50-10	36-24	59-1	46-14	50-9-1	52-8	56-4	43-17	57-3	60-0	60-0	60-0
Marmot	604-415-1	5-55	8-52	15-45	9-51	9-51	10-50		31-29	50-10	46-14	48-11-1	52-8	53-7	44-16	53-7	57-3	57-3	57-3
JKU-CODA	566-454	3-57	9-51	3-57	4-56	13-47	24-36	29-31		25-35	34-26	51-9	54-6	50-10	35-25	58-2	59-1	57-3	58-2
DynamicEntropy	492-528	1-59	2-58	4-56	9-51	0-60	1-59	10-50	35-25		26-34	32-28	42-18	56-4	37-23	59-1	60-0	59-1	59-1
SomeRegret	485-535	4-56	5-55	14-46	2-58	8-52	14-46	14-46	26-34	34-26		42-18	24-36	43-17	34-26	48-12	57-3	58-2	58-2
trout	426-589-5	1-59	0-60	4-56	0-60	0-60	9-50-1	11-48-1	9-51	28-32	18-42		40-18-2	37-23	32-28	57-2-1	60-0	60-0	60-0
attacker	417-601-2	0-60	1-59	6-54	0-60	14-46	8-52	8-52	6-54	18-42	36-24	18-40-2		32-28	36-24	54-6	60-0	60-0	60-0
GarrisonNRL	371-649	0-60	3-57	7-53	3-57	1-59	4-56	7-53	10-50	4-56	17-43	23-37	28-32		37-23	47-13	60-0	60-0	60-0
uccchess	357-659-4	7-52-1	8-49-3	12-48	23-37	7-53	17-43	16-44	25-35	23-37	26-34	28-32	24-36	23-37		25-35	25-35	35-25	33-27
random	263-756-1	0-60	0-60	2-58	0-60	0-60	3-57	7-53	2-58	1-59	12-48	2-57-1	6-54	13-47	35-25		60-0	60-0	60-0
arandombot	101-919	0-60	0-60	0-60	0-60	0-60	0-60	3-57	1-59	0-60	3-57	0-60	0-60	0-60	35-25	0-60		29-31	30-30
srcork	97-923	0-60	0-60	0-60	0-60	0-60	0-60	3-57	3-57	1-59	2-58	0-60	0-60	0-60	25-35	0-60	31-29		32-28
uccch	94-926	0-60	0-60	1-59	0-60	0-60	0-60	3-57	2-58	1-59	2-58	0-60	0-60	0-60	27-33	0-60	30-30	28-32	

Figure 1: Crosstable of wins each bot had against each other bot in the competition.

Fianchetto performed better against Kevin, Kevin performed better against Châteaux, and Châteaux performed better against Fianchetto.

Ignoring StrangeFish2, this cycle in performance illustrates the exploitability one can have in a seemingly strong algorithm by an adaptive opponent in games with imperfect information. E.g., although Fianchetto's strategy seems strong against Kevin, clearly if Kevin adapted its strategy (to be more like Châteaux's, for example) after playing Fianchetto, it would likely win. Adapting to a particular opponent is an opportunity for greater performance in imperfect information games, but it is also, of course, a risk as it opens one up to potential exploitability (compared to an equilibrium strategy). Table 2 notes bots that attempt to model their specific opponent in some way (*Uses opponent specific models* column).[1] Although modeling or adapting to a bot's specific opponent is not typical among the best-performing bots, we expect this is incidental and due to a lack of resources to dedicate to that effort at the current time. As the research advances, we expect strategies closer to equilibrium to arise and the value of opponent modeling to diminish. However, in the shorter term, specific-opponent modeling may become central to top bots, particularly with all the game logs available online.

To provide some context over a few years, Oracle, now ranked 6, was the strongest running bot before the NeurIPS competitions[2] and was ranked 3rd in the 2019 NeurIPS competition. Châteaux, previously known as penumbra, won the 2020 online competition held by the Johns Hopkins University Applied Physics Laboratory. Fianchetto won in NeurIPS 2021. Figure 2 illustrates Bayes Elo ratings of each bot with uncertainty (including forfeits in the computation). Figure 3 illustrates the evolution of the Elo ratings as the tournament progressed omitting forfeits (which may be caused by code crashes, computer crashes, connection problems, or other issues unrelated to algorithm).[3] One can see that

1. One's opponent's name is available in RBC.
2. Prior to hosting NeurIPS competitions, the organizers had one bot that outperformed Oracle head to head although it was not maintained and we do not expect it was notably stronger than Oracle.
3. The Elo values in the two figures do not align since removing forfeits from the computation affects the values for all bots.

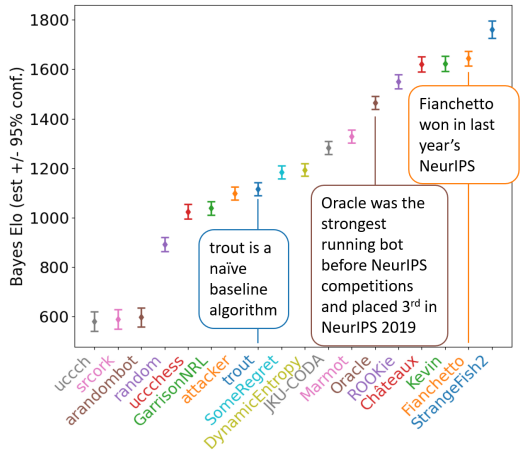

Figure 2: Bayes Elo ratings of each bot with a 95% confidence interval.

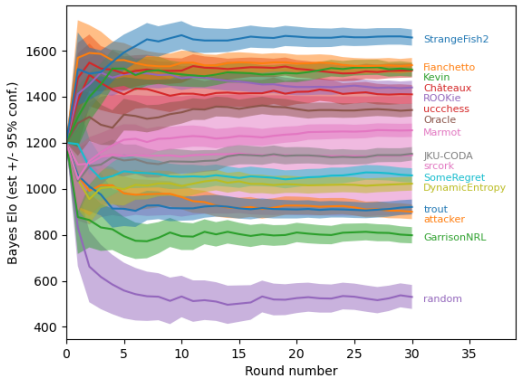

Figure 3: Bayes Elo ratings and 95% confidence intervals as the tournament progressed, excluding forfeits.

several bots' ratings were significantly decreased by forfeits like uccchess, SomeRegret, and srcork.

Uncertainty Management: We have historically asked how raw uncertainty is related to performance. Specifically we measure uncertainty in terms of the number of board states that are possible given all of a player's observations up to a point in the game. This measurement is far from comprehensive because it captures no notion of probability, but it is something we can compute and gives a general quantitative sense for the amount of uncertainty. (Ten states could, for example, represent less uncertainty than two, if one of those ten states was the true state with .99 probability while the two states were equally likely. Another example limitation of this metric is a bot could appear to have more or less uncertainty because it tends to have longer or shorter games.)

The "Median # Board States" column in Table 3 contains the median number of board states possible from each bots' observations after the bot senses, each turn of the game. We see that the winner, StrangeFish2 is only ranked 10 on this metric of uncertainty and that the median number of possible board states for StrangeFish2 is twice that of some bots, among the highest of the stronger bots. More generally, while the top bots all appear to keep this number under 35, beyond that, the number-of-possible-board-states rank does not appear to be strongly correlated with tournament rank. This limited correlation indicates that simply minimizing uncertainty is insufficient. Accounting for the impact of exactly what one will do with an observation appears to be critical, in alignment with StrangeFish2's primary upgrade from previous years.

We note that the number of board states is *not* the same as the number of game states or histories, which includes all of both players' senses (or belief states) and move history, all of which potentially impact optimal strategy. The number of possible game states inevitably grows exponentially each turn due to the private sense action.

Comparison to a Chess Engine: We also compare the tournament RBC bots' moves to moves that would be strong in traditional chess, specifically by comparing to the moves

recommended by Stockfish (an available superhuman chess engine). The "Engine Move Agreement" column in Table 3 displays the portion of time each bot's selected move was one of the top three moves recommended by Stockfish. Because RBC also includes board states that are invalid in chess, we considered any moves that capture the opponent's king to be in the top three recommended by the engine when possible.

Like with the uncertainty, while we see most of the top bots keep this metric above some threshold, roughly 50%, it is not a clear indicator of performance. StrangeFish2 only ranked 6 in chess-engine move agreement, for example. The lack of a stronger correlation here could result from several intuitive notions. Bots may benefit from being more aggressive and risk-taking than in standard chess (e.g., placing pieces where they would be less strategically located if seen but enable attacks if temporarily unseen). Another possibility is that bots benefit from moving more conservatively. For example, the optimal strategy in RBC may be to frequently protect against attacks that are not actually possible given the true board state.

We observe that Marmot and DynamicEntropy have especially low move agreement among the top 10 bots. We hypothesize this results from the approaches accounting for uncertainty, both explicitly keeping track of information sets in their search algorithms, which starkly contrasts a perfect-information chess engine although they both use chess engines on leaf nodes.

Table 3: A comparison of each bot's tournament rank to their relative uncertainty management and use of traditional chess moves.

Bot	Elo	Elo rank	Median # Board States	# Board States Rank	# Board States Δ Rank	Engine Move Agreement	Move Agree. Rank	Move Agree. Δ Rank
StrangeFish2	1762	1	32	10	+9	62%	6	+5
Fianchetto	1644	2	20	4	+2	69%	3	+1
Kevin	1623	3	16	2	-1	74%	1	-2
Châteaux	1621	4	21	5	+1	53%	9	+5
ROOKie	1551	5	29	8	+3	72%	2	-3
Oracle	1465	6	16	2	-4	67%	4	-2
Marmot	1329	7	28	7	0	32%	11	+4
JKU-CODA	1283	8	29	8	0	55%	8	0
DynamicEntropy	1194	9	34	11	+2	29%	13	+4
SomeRegret	1184	10	13	1	-9	67%	5	-5
trout	1116	11	214	13	+2	44%	10	-1
attacker	1099	12	11356	15	+3	5%	15	+3
GarrisonNRL	1039	13	46	12	-1	31%	12	-1
Uccchess	1025	14	21	5	-9	58%	7	-7
random	893	15	2278	14	+1	8%	14	-1

7. Conclusions

The NeurIPS 2022 machine RBC competition brought numerous people together to attempt to create better methods for making strategic decisions with significant private information and little common knowledge. We continue to see a wider diversity of algorithms, and they are improving in general. Conducting practical and sound search in settings like RBC remains an open problem. We hope to hold future competitions to make progress toward this goal until such an algorithm has been convincingly identified.

References

Timo Bertram, Johannes Fürnkranz, and Martin Müller. Predicting human card selection in Magic: The Gathering with contextual preference ranking. In *IEEE Conference on Games (CoG)*, 2021.

Noam Brown and Tuomas Sandholm. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, 359(6374):418–424, 2017a.

Noam Brown and Tuomas Sandholm. Safe and nested subgame solving for imperfect-information games. In *Neural Information Processing Systems (NIPS)*, 2017b.

Noam Brown and Tuomas Sandholm. Superhuman AI for multiplayer poker. *Science*, 365 (6456):885–890, 2019.

Noam Brown, Tuomas Sandholm, and Brandon Amos. Depth-limited solving for imperfect-information games. In *Neural Information Processing Systems (NeurIPS) Deep Reinforcement Learning Workshop (DRL)*, 2018.

Noam Brown, Adam Lerer, Sam Gross, and Tuomas Sandholm. Deep counterfactual regret minimization. In *International Conference on Machine Learning (ICML)*, 2019.

Noam Brown, Anton Bakhtin, Adam Lerer, and Qucheng Gong. Combining deep reinforcement learning and search for imperfect-information games. In *Neural Information Processing Systems (NeurIPS)*, 2020.

Cameron B. Browne, Edward Powley, Daniel Whitehouse, Simon M. Lucas, Peter I. Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):1–43, 2012.

Neil Burch, Michael Johanson, and Michael Bowling. Solving imperfect information games using decomposition. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2014.

Murray Campbell, A. Joseph Hoane Jr, and Feng-hsiung Hsu. Deep Blue. *Artificial Intelligence*, 134(1-2):57–83, 2002.

Paolo Ciancarini and Gian Piero Favini. Monte Carlo tree search in Kriegspiel. *Artificial Intelligence*, 174(11):670–684, 2010.

Gregory Clark. Deep synoptic Monte-Carlo planning in Reconnaissance Blind Chess. In *Neural Information Processing Systems (NeurIPS)*, 2021.

Rémi Coulom. Bayesian elo rating, February 2006. URL https://www.remi-coulom.fr/Bayesian-Elo/.

Peter I. Cowling, Edward J. Powley, and Daniel Whitehouse. Information set Monte Carlo tree search. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(2): 120–143, 2012.

Ryan W. Gardner, Corey Lowman, Casey Richardson, Ashley J. Llorens, Jared Markowitz, Nathan Drenkow, Andrew Newman, Gregory Clark, Gino Perrotta, Robert Perrotta, Timothy Highley, Vlad Shcherbina, William Bernadoni, Mark Jordan, and Asen Asenov. The first international competition in machine Reconnaissance Blind Chess. *Proceedings of Machine Learning Research (PMLR)*, 123, 2020.

Johannes Heinrich and David Silver. Deep reinforcement learning from self-play in imperfect-information games. In *Neural Information Processing Systems (NeurIPS) Deep Reinforcement Learning Workshop (DRL)*, 2016.

Johannes Heinrich, Marc Lanctot, and David Silver. Fictitious self-play in extensive-form games. In *International Conference on Machine Learning (ICML)*, 2015.

Levente Kocsis and Csaba Szepesvári. Bandit based Monte-Carlo planning. In *European Conference on Machine Learning (ECML)*, 2006.

Viliam Lisý, Marc Lanctot, and Michael Bowling. Online Monte Carlo counterfactual regret minimization for search in imperfect information games. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2015.

Stephen McAleer, Gabriele Farina, Marc Lanctot, and Tuomas Sandholm. ESCHER: Eschewing importance sampling in games by computing a history value function to estimate regret. In *Neural Information Processing Systems (NeurIPS)*, 2022.

Reid McIlroy-Young, Siddhartha Sen, Jon Kleinberg, and Ashton Anderson. Aligning superhuman AI with human behavior: Chess as a model system. In *ACM Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2020.

Matej Moravčík, Martin Schmid, Neil Burch, Viliam Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. DeepStack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337):508–513, 2017.

Andrew J. Newman, Casey L. Richardson, Sean M. Kain, Paul G. Stankiewicz, Paul R. Guseman, Blake A. Schreurs, and Jeffrey A. Dunne. Reconnaissance blind multi-chess: An experimentation platform for ISR sensor fusion and resource management. In *Signal Processing, Sensor/Information Fusion, and Target Recognition XXV*, volume 9842, 2016.

Julien Perolat, Bart De Vylder, Daniel Hennes, Eugene Tarassov, Florian Strub, Vincent de Boer, Paul Muller, Jerome T. Connor, Neil Burch, Thomas Anthony, Stephen McAleer, Romuald Elie, Sarah H. Cen, Zhe Wang, Audrunas Gruslys, Aleksandra Malysheva, Mina Khan, Sherjil Ozair, Finbarr Timbers, Toby Pohlen, Tom Eccles, Mark Rowland, Marc Lanctot, Jean-Baptiste Lespiau, Bilal Piot, Shayegan Omidshafiei, Edward Lockhart, Laurent Sifre, Nathalie Beauguerlange, Remi Munos, David Silver, Satinder Singh, Demis Hassabis, and Karl Tuyls. Mastering the game of Stratego with model-free multiagent reinforcement learning. *Science*, 378(6623):990–996, 2022.

Gino Perrotta and Robert Perrotta. StrangeFish, October 2019. URL https://github.com/ginop/reconchess-strangefish.

Gino Perrotta, Ryan W. Gardner, Mohammad Taufeeque, Nitish Tongia, Shivram Kalyanakrishnan, Gregory Clark, Kevin Wang, Eitan Rothberg, Brady P. Garrison, Prithviraj Dasgupta, Callum Canavan, and Lucas McCabe. The second NeurIPS tournament of Reconnaissance Blind Chess. *Proceedings of Machine Learning Research*, 176, 2022.

Stuart Russell and Jason Wolfe. Efficient belief-state AND-OR search, with application to Kriegspiel. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2005.

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–489, 2016.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of Go without human knowledge. *Nature*, 550: 354–359, 2017.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 362(6419):1140–1144, 2018.

Eric Steinberger, Adam Lerer, and Noam Brown. DREAM: deep regret minimization with advantage baselines and model-free learning. In *Neural Information Processing Systems (NeurIPS)*, 2020.

Michal Sustr, Vojtech Kovarík, and Viliam Lisý. Monte Carlo continual resolving for online strategy computation in imperfect information games. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2019.

Brian Hu Zhang and Tuomas Sandholm. Subgame solving without common knowledge. In *Neural Information Processing Systems (NeurIPS)*, 2021.

Martin Zinkevich, Michael Johanson, Michael Bowling, and Carmelo Piccione. Regret minimization in games with incomplete information. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2008.