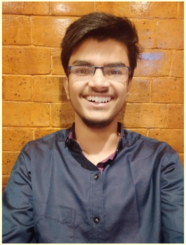
On Designing a Winning Agent for Reconnaissance Blind Chess (RBC)

Shivaram Kalyanakrishnan

Department of Computer Science and Engineering
Indian Institute of Technology Bombay

February 2024

Contributors



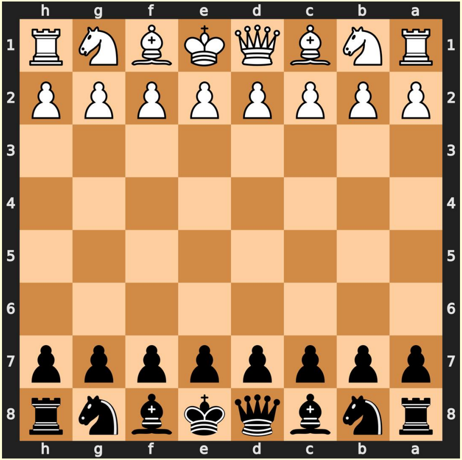
Mohammad Taufeeque

Nitish Tongia

Anvay Shah

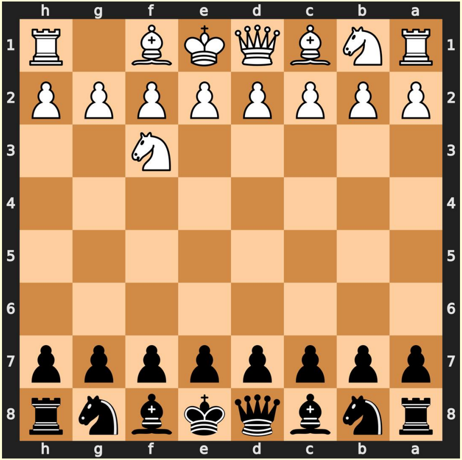
Puranjay Datta

Chess



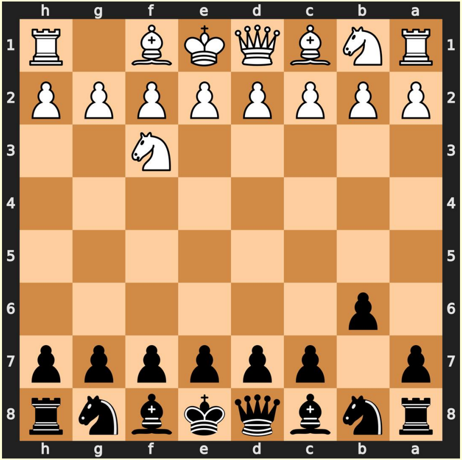
Starting position

Chess



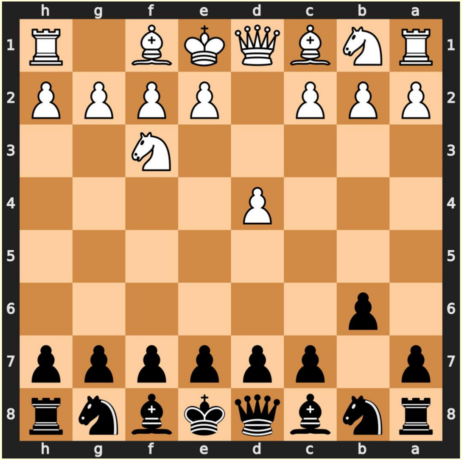
White plays g1f3

Chess



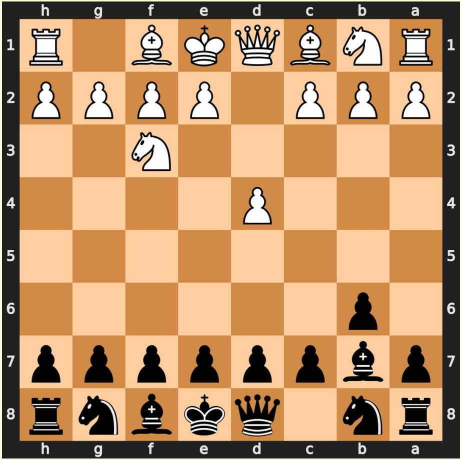
Black plays b7b6

Chess



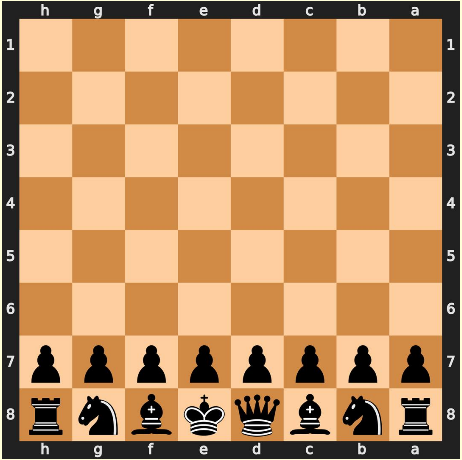
White plays d2d4

Chess



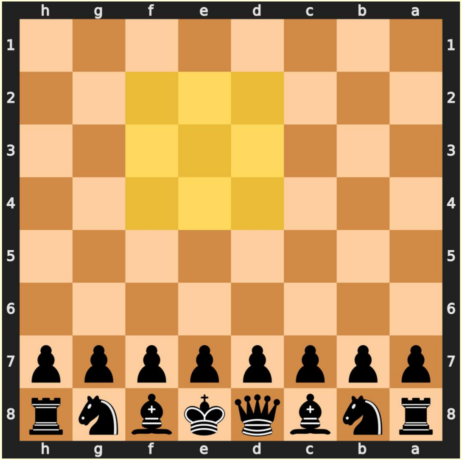
Black plays c8b7

RBC



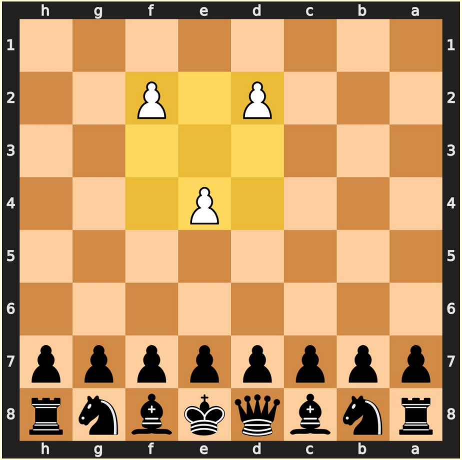
Starting position of RBC

RBC



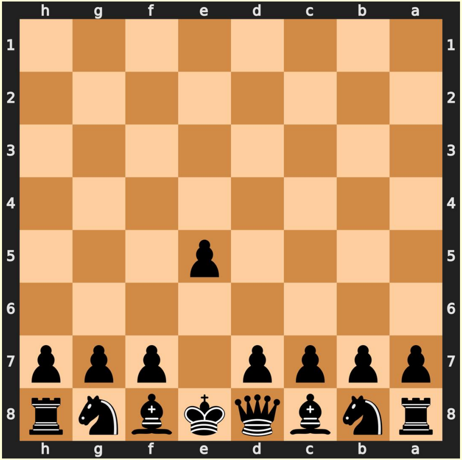
Black chooses sense region e3

RBC



Information gained from sense

RBC



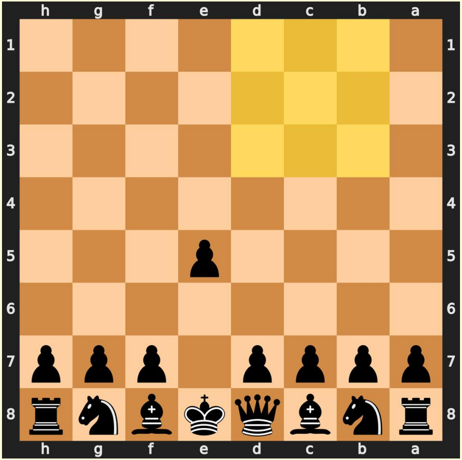
Black plays e7e5

RBC

Where should black sense next?

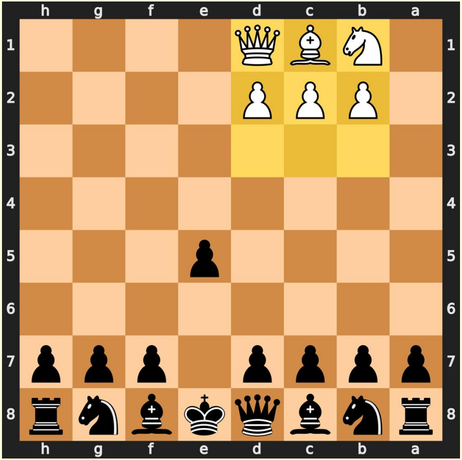
Given current *history*: sense e3, observe Pe4, move e7e5

RBC



Black chooses sense region c2

RBC



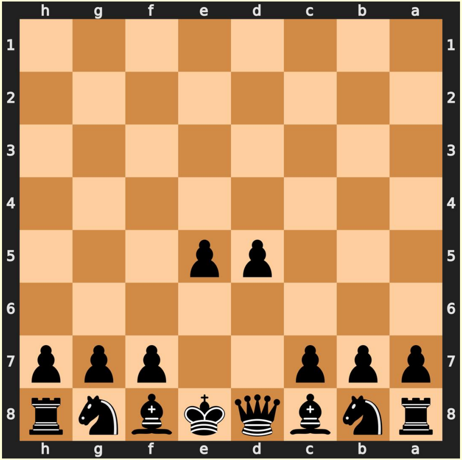
No new information gained from sense!

RBC

Where should black move next?

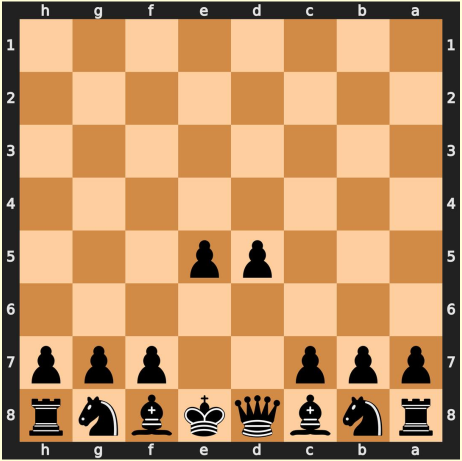
Given current history: sense e3, observe Pe4, move e7e5, sense c2, observe no new information

RBC



Black plays d7d5

RBC



Black plays d7d5

Game proceeds similarly. Additional RBC details: captures, null moves.

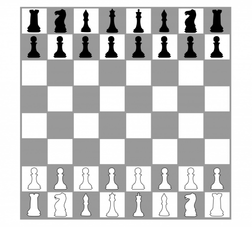
Outline

1. Challenges of RBC
2. Baseline agent: StrangeFish
3. Our agent: Fianchetto
4. Competition results and analysis
5. Conclusion

Outline

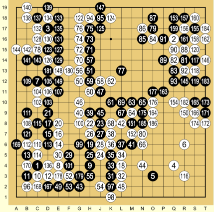
1. Challenges of RBC
2. Baseline agent: StrangeFish
3. Our agent: Fianchetto
4. Competition results and analysis
5. Conclusion

Board Games and AI



Chess (CHH02) [1]

Backgammon (T94) [2]



Lee Sedol (B) vs AlphaGo (W) - Game 1

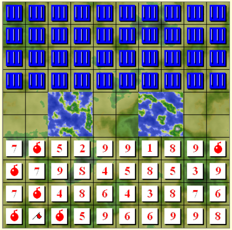
Go (S+16) [3]

[1] `https://www.publicdomainpictures.net/en/view-image.php?image=55671&picture=backgammon`
[2] `https://www.publicdomainpictures.net/pictures/80000/velka/chess-board-and-pieces.jpg`.
[3] `https://upload.wikimedia.org/wikipedia/commons/thumb/5/56/Lee_Sedol_%28B%29_vs_AlphaGo_%28W%29_-_Game_1.svg/734px-Lee_Sedol_%28B%29_vs_AlphaGo_%28W%29_-_Game_1.svg.png`. CC image courtesy of Wesalius on WikiMedia Commons licensed under CC-BY-SA-4.0.

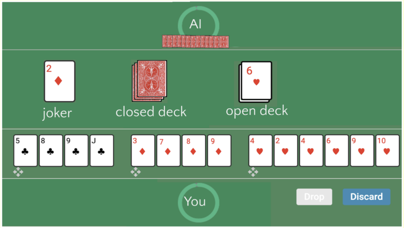
Games of Imperfect Information

Scrabble (S02) [1]

Poker (M+17,BS19) [2]



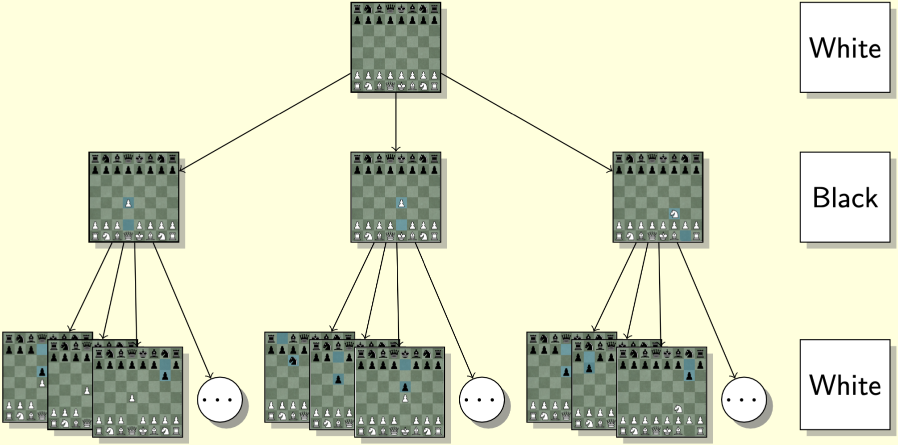
Stratego (P+22) [3]



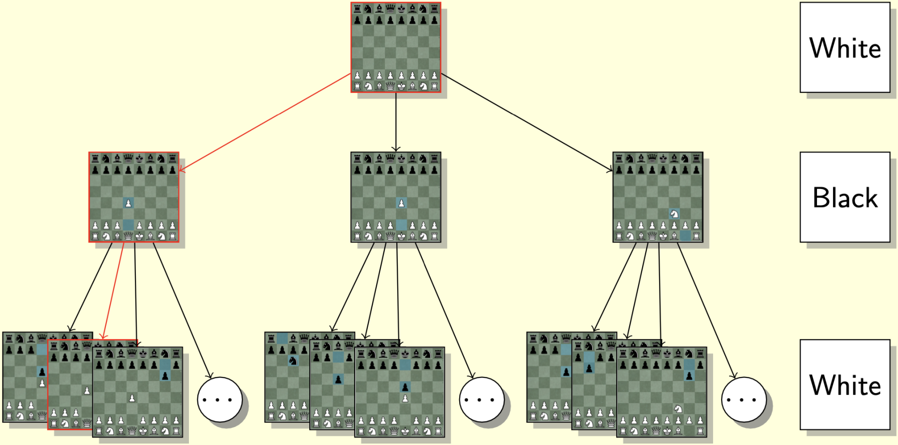
Rummy

[1] `https://www.publicdomainpictures.net/en/free-download.php?image=scrabble-board&id=53283`.
[2] `https://www.publicdomainpictures.net/en/free-download.php?image=poker&id=84950`
[3] `https://upload.wikimedia.org/wikipedia/commons/0/05/Stratego.png`. CC image courtesy of Andreas Kaufmann on WikiMedia Commons licensed under CC-BY-SA-3.0.

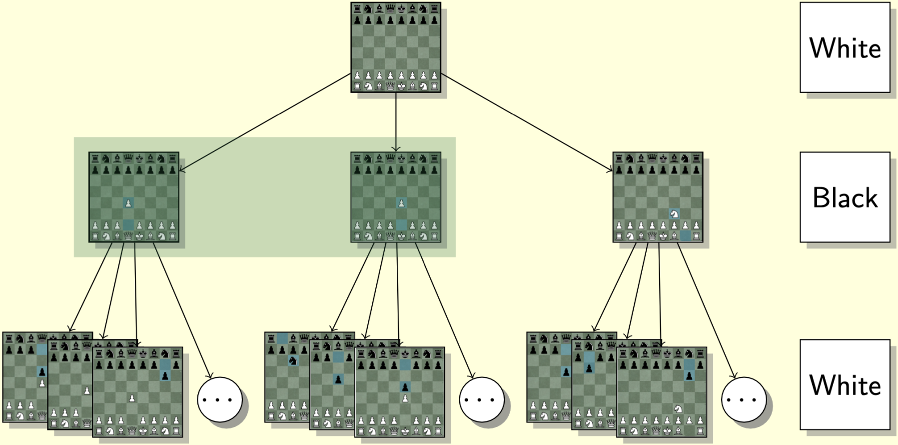
Challenge: States → Information sets



Challenge: States $\rightarrow$ Information sets



Challenge: States → Information sets



Why is RBC Especially Hard?

- Private versus public/shared information.
  Almost all information in RBC is private.

Why is RBC Especially Hard?

- Private versus public/shared information.
  Almost all information in RBC is private.

- Game horizon.
  On average 50–60 in RBC between good players; can go into 100s.

Why is RBC Especially Hard?

- Private versus public/shared information.
  Almost all information in RBC is private.
- Game horizon.
  On average 50–60 in RBC between good players; can go into 100s.
- How many opponent histories aliased into agent's information set?
  Estimated 10^{68} for RBC (MGL18); 10^{4} in Poker.

Why is RBC Especially Hard?

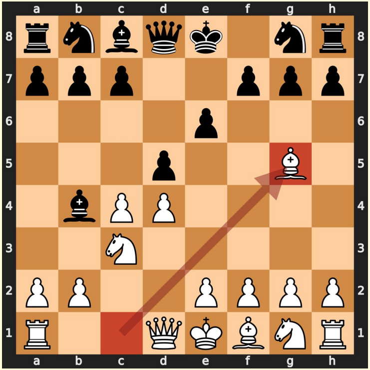
- Private versus public/shared information.
  Almost all information in RBC is private.

- Game horizon.
  On average 50–60 in RBC between good players; can go into 100s.

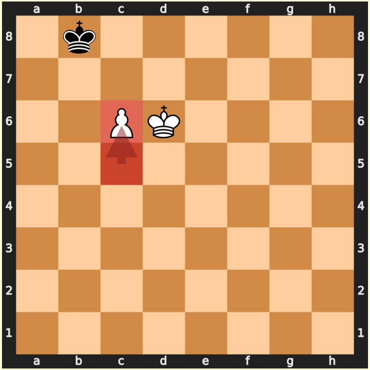
- How many opponent histories aliased into agent's information set?
  Estimated 10^{68} for RBC (MGL18); 10^4 in Poker.

- But Chess is so well-understood. Does that help?

RBC: Like Chess and also Unlike Chess



Potentially successful move in RBC, bad in Chess.

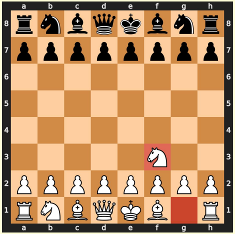


Good move in Chess, bad in RBC.

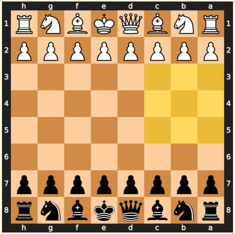
Outline

1. Challenges of RBC
2. Baseline agent: StrangeFish
3. Our agent: Fianchetto
4. Competition results
5. Conclusion

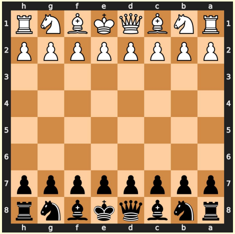
Primitive: Maintaining a Board Set B



Before sense: $|B| = 21$

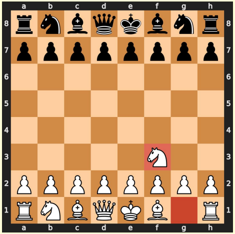


Sense action

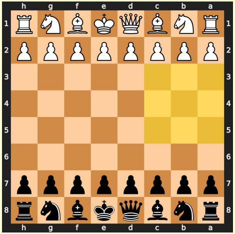


After sense: $|B| = 13$

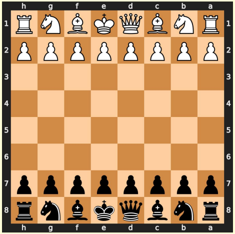
Primitive: Maintaining a Board Set B



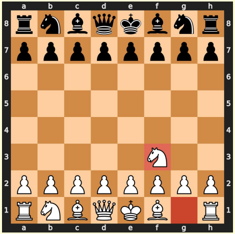
Before sense: $|B| = 21$



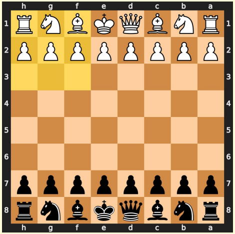
Sense action



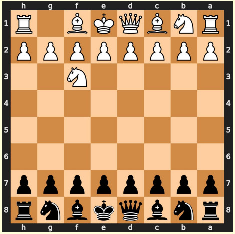
After sense: $|B| = 13$



Before sense: $|B| = 21$

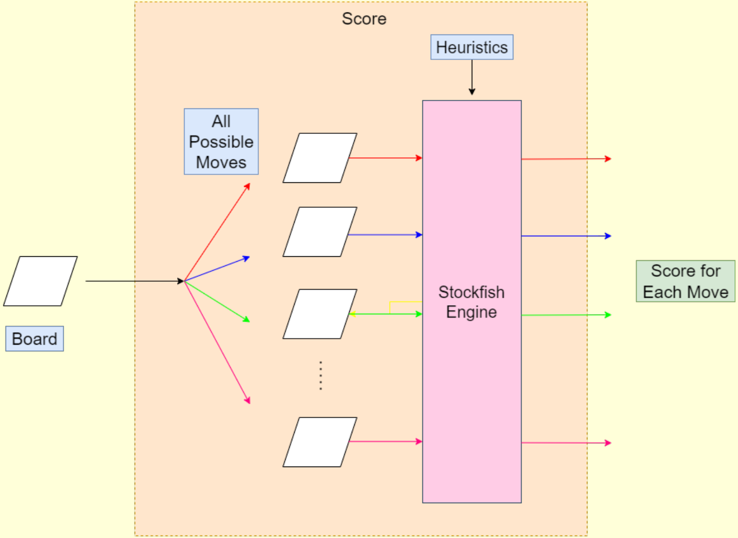


Sense action



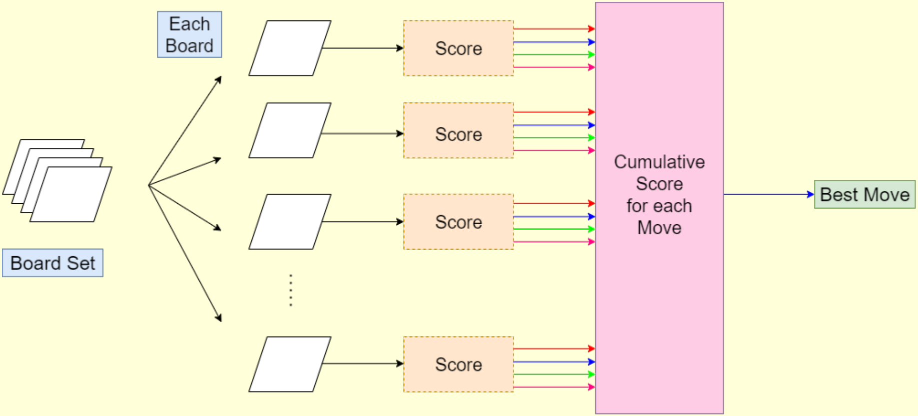
After sense: $|B| = 1$

StrangeFish 2019 Baseline



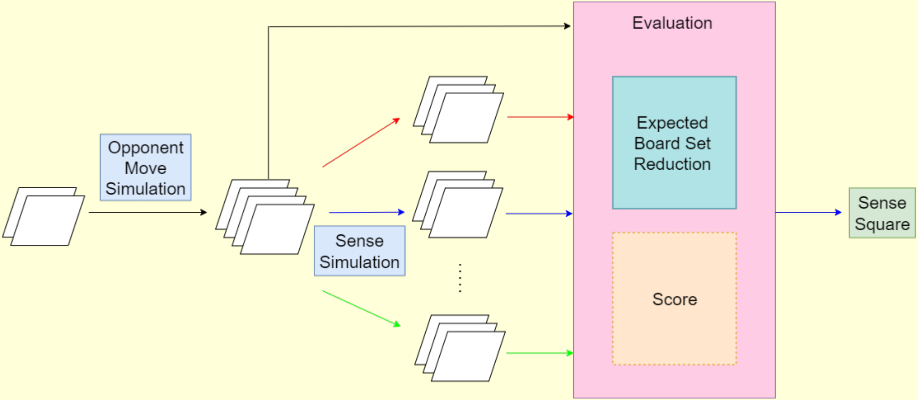
Score function takes a single board as input and provides the score for each move on this board.

StrangeFish 2019



Moving strategy: choose the move that maximises an aggregate score combining (weighted) mean, min, and max scores over *B*.

StrangeFish 2019

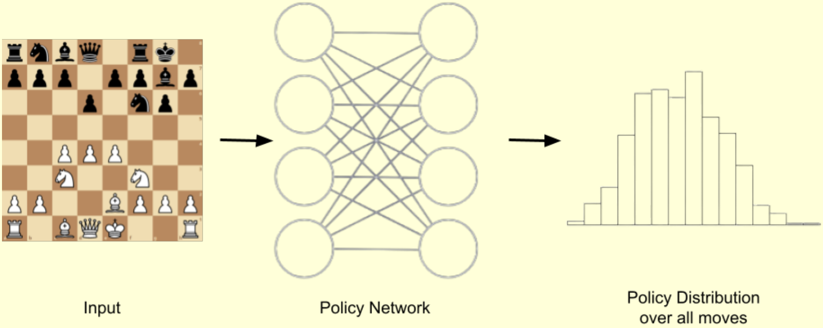


Sensing strategy: choose a sense square to maximise a combination of board set reduction and potential change in value.

Outline

Fianchetto v1: Using LC0 instead of Stockfish

- Leela Chess Zero (LC0) (PL21) neural network instead of StockFish (RCK21) for faster evaluation.



Fianchetto v1: Using LC0 instead of Stockfish

- Achieves $\approx$ 30x speedup of no. of boards evaluated/sec through batching

	Time per engine call (s)	Effective no. of boards evaluated / s	
		Without batching	With batching
Stockfish	0.005	200	3200 (16 Threads)
Lc0 (1GB GPU)	0.321	93	95232 (batch size = 1024)

Table: Comparison of throughput of Stockfish and Lc0, performed on a desktop machine with Intel Core i5-4690 CPU@3.50GHz and Nvidia GeForce GTX 980 GPU.

Fianchetto v1: Using LC0 instead of Stockfish

- Achieves $\approx$ 30x speedup of no. of boards evaluated/sec through batching

	Time per engine call (s)	Effective no. of boards evaluated / s Without batching	 With batching
Stockfish	0.005	200	3200 (16 Threads)
Lc0 (1GB GPU)	0.321	93	95232 (batch size = 1024)

Table: Comparison of throughput of Stockfish and Lc0, performed on a desktop machine with Intel Core i5-4690 CPU@3.50GHz and Nvidia GeForce GTX 980 GPU.

Fianchetto v1: Using LC0 instead of Stockfish

- Risk of overfitting to Chess!

	Small n/w	Medium n/w	Large n/w
Chess Rating	1416	1453	1572
RBC Rating	1248	1502	1350

Table: Comparison of ratings of different-sized Lc0 networks.

Fianchetto v1: Using LC0 instead of Stockfish

- Risk of overfitting to Chess!

	Small n/w	Medium n/w	Large n/w
Chess Rating	1416	1453	1572
RBC Rating	1248	1502	1350

Table: Comparison of ratings of different-sized Lc0 networks.

Fianchetto v2: Persistent Board Belief

- Belief state b is a weight/probability distribution over S.
- Critical for intelligent decision making.
- At any stage, support of b (all reachable states) can be calculated exactly.

Fianchetto v2: Persistent Board Belief

- Belief state b is a weight/probability distribution over S.
- Critical for intelligent decision making.
- At any stage, support of b (all reachable states) can be calculated exactly.
- Under StrangeFish: $b(s) \propto \text{sigmoid}(\text{StockFishScore}(s))$.

Fianchetto v2: Persistent Board Belief

- Belief state b is a weight/probability distribution over S.
- Critical for intelligent decision making.
- At any stage, support of b (all reachable states) can be calculated exactly.
- Under StrangeFish: $b(s) \propto$ sigmoid(StockFishScore(s)).
- Ignores previous belief, opponent's policy.

Fianchetto v2: Persistent Board Belief

- Belief state b is a weight/probability distribution over S.
- Critical for intelligent decision making.
- At any stage, support of b (all reachable states) can be calculated exactly.
- Under StrangeFish: $b(s) \propto \text{sigmoid(StockFishScore(s))}$.
- Ignores previous belief, opponent's policy.

- Suppose b is belief state before Fianchetto's move, and Fianchetto plays move a. Then by basic probability, the belief state b' after the move is:

$$b'(s') = \sum_{s} b(s)\mathbf{1}[(s, a) \text{ leads to } s'].$$

Fianchetto v2: Persistent Board Belief

- Belief state b is a weight/probability distribution over S.
- Critical for intelligent decision making.
- At any stage, support of b (all reachable states) can be calculated exactly.
- Under StrangeFish: $b(s) \propto \text{sigmoid}(\text{StockFishScore}(s))$.
- Ignores previous belief, opponent's policy.

- Suppose b is belief state before Fianchetto's move, and Fianchetto plays move a. Then by basic probability, the belief state b' after the move is:

$$b'(s') = \sum_{s} b(s)\mathbf{1}[(s,a) \text{ leads to } s'].$$

- Let b'' be the belief state after the opponent has played, Fianchetto has sensed observation z. By Bayes' Rule,

$$b''(s') = \mathbb{P}\{s'|b',z\} = \sum_{a} \mathbb{P}\{s'|b',a,z\} \sum_{s} \mathbb{P}\{a|s\}\mathbb{P}\{s|b'\};$$

$$\mathbb{P}\{s'|b',a,z\} \propto \mathbb{P}\{z|s',a\} \sum_{s} \mathbb{P}\{s'|s,a\}\mathbb{P}\{s|b'\}.$$

Fianchetto v2: Persistent Board Belief

- Belief state b is a weight/probability distribution over S.
- Critical for intelligent decision making.
- At any stage, support of b (all reachable states) can be calculated exactly.
- Under StrangeFish: $b(s) \propto$ sigmoid(StockFishScore(s)).
- Ignores previous belief, opponent's policy.

- Suppose b is belief state before Fianchetto's move, and Fianchetto plays move a. Then by basic probability, the belief state b' after the move is:

$$b'(s') = \sum_{s} b(s)\mathbf{1}[(s, a) \text{ leads to } s'].$$

- Let b'' be the belief state after the opponent has played, Fianchetto has sensed observation z. By Bayes' Rule,

$$b''(s') = \mathbb{P}\{s'|b', z\} = \sum_{a} \mathbb{P}\{s'|b', a, z\} \sum_{s} \mathbb{P}\{a|s\}\mathbb{P}\{s|b'\};$$

$$\mathbb{P}\{s'|b', a, z\} \propto \mathbb{P}\{z|s', a\} \sum_{s} \mathbb{P}\{s'|s, a\}\mathbb{P}\{s|b'\}.$$

- But what is $\mathbb{P}\{a|s\}$ (the opponent model)?

Fianchetto v2: Persistent Board Belief

- Belief state b is a weight/probability distribution over S.
- Critical for intelligent decision making.
- At any stage, support of b (all reachable states) can be calculated exactly.
- Under StrangeFish: $b(s) \propto \text{sigmoid(StockFishScore(s))}$.
- Ignores previous belief, opponent's policy.

- Suppose b is belief state before Fianchetto's move, and Fianchetto plays move a. Then by basic probability, the belief state b' after the move is:

$$b'(s') = \sum_{s} b(s)\mathbf{1}[(s, a) \text{ leads to } s'].$$

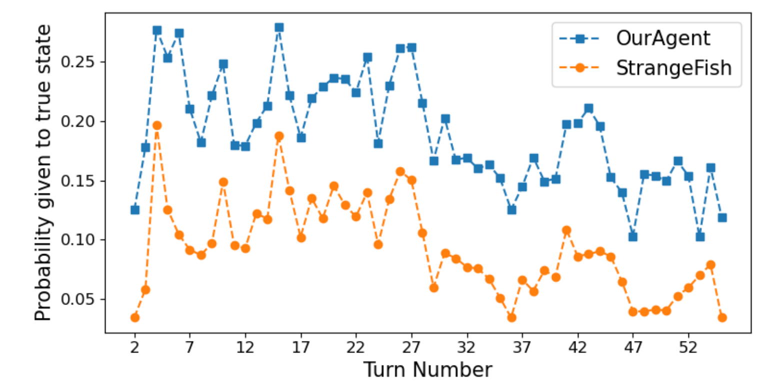
- Let b'' be the belief state after the opponent has played, Fianchetto has sensed observation z. By Bayes' Rule,

$$b''(s') = \mathbb{P}\{s'|b', z\} = \sum_{a} \mathbb{P}\{s'|b', a, z\} \sum_{s} \mathbb{P}\{a|s\}\mathbb{P}\{s|b'\};$$

$$\mathbb{P}\{s'|b', a, z\} \propto \mathbb{P}\{z|s', a\} \sum_{s} \mathbb{P}\{s'|s, a\}\mathbb{P}\{s|b'\}.$$

- But what is $\mathbb{P}\{a|s\}$ (the opponent model)?
We assume the opponent plays according to LC0!

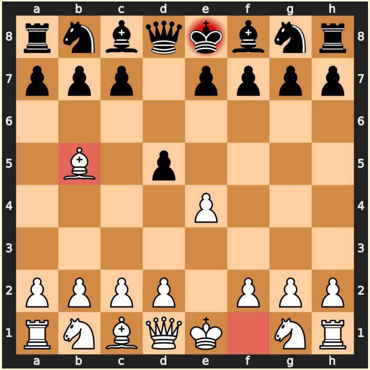
Fianchetto v2: Persistent Board Belief



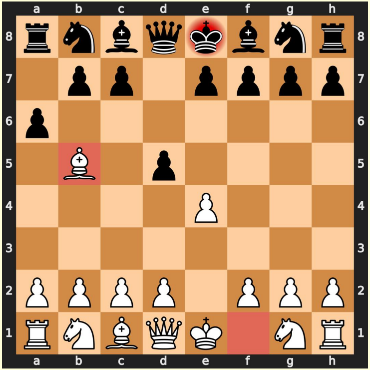
Probability associated with true state by Fianchetto (v2) and StrangeFish

Fianchetto v3: RBC-Specific Incentives

- Supplement LC0 evaluation to promote RBC-specific “sneak attacks”.



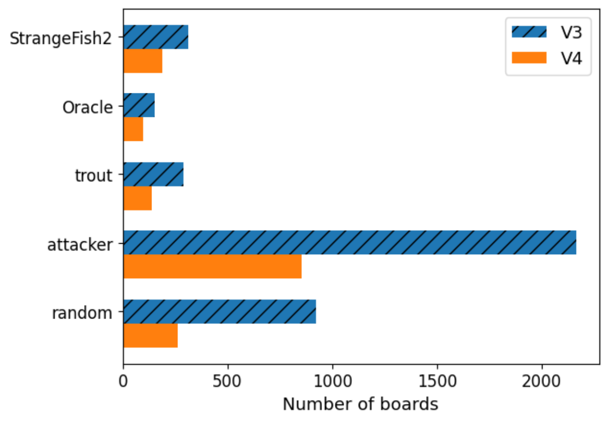
Low risk, large incentive.



High risk, small incentive.

Fianchetto v4: Board Set Size

- Add dynamically weighted uniform dist. to opponent's move probabilities.
- Adjust weightage of expected board set reduction in sense strategy.



Average information set size in games played on the RBC server.

Outline

NeurIPS 2021 Tournament

- Round robin tournament between 18 bots
- Pairwise 60 games (equally split as black and white)
- Positive winning record of atleast 66% against every other bot in the tournament
- Dominant performance with overall win ratio above 90%

`Fianchetto` (1759)	Score
`StrangeFish2` (1662)	41-19
`penumbra` (1584)	40-20
`Kevin` (1544)	40-20
`Oracle` (1503)	51-9
`Gnash` (1454)	49-11
`Marmot` (1315)	56-4
`DynamicEntropy` (1299)	59-1
`wbernar5` (1219)	58-2
`Frampt` (1208)	59-1
`GarrisonNRL` (1140)	59-1
`trout` (1127)	59-1
`callumcanavan` (1066)	60-0
`attacker` (1049)	60-0
`URChIn` (854)	60-0
`armandli` (777)	60-0
`random` (753)	60-0
`ai_games_cvi` (288)	60-0
Overall	**931-89**

Table: NeurIPS 2021 RBC Tournament results

Intermediate Versions (evaluated post-competition)

Version	V0	SF2	Or	tr	att	ran	Overall
V0	30-30	16-44	39-21	57-3	56-4	60-0	258-102
V1	24-36	13-47	36-24	58-2	50-10	59-1	240-120
V2	39-21	35-25	46-14	59-1	58-2	60-0	297-63
V3	48-12	32-28	47-13	59-1	58-2	59-1	303-57
V4	48-12	35-25	47-13	59-1	60-0	60-0	309-51

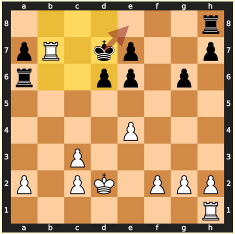
Table: Win-loss scores from a 60-game match between row agent and column agent. V0 is the same as StrangeFish; its row is populated using its last 60 games in a specified window in November-December 2021 on the RBC server. The column for V0 is obtained locally, whereas all other columns (SF2 = `StrangeFish2`, Or = `Oracle`, tr = `trout`, att = `attacker`, ran = `random`) are obtained from games played on the server.

NeurIPS 2022 Tournament

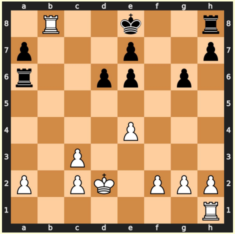
StrangeFish2 (1762)	21-34-5
Fianchetto (1644)	Score
Kevin (1623)	31-0-29
Chateaux (1621)	18-0-42
ROOKie (1551)	37-18-5
Oracle (1465)	49-9-2
Marmot (1329)	52-0-8
JKU-CODA (1283)	51-0-9
DynamicEntropy (1194)	58-0-2
SomeRegret (1184)	55-0-5
trout (1116)	60-0-0
attacker (1099)	59-0-1
GarrisonNRL (1039)	57-0-3
ucchess (1025)	49-8-3
random (893)	60-0-0
arandombot (598)	60-0-0
srcork (590)	60-0-0
uccch (581)	60-0-0
Overall	**837-168-15**

Table: NeurIPS 2022 RBC Tournament results

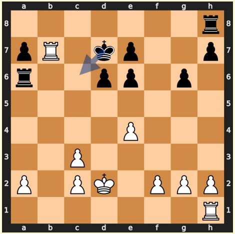
Blunders!



Played Move

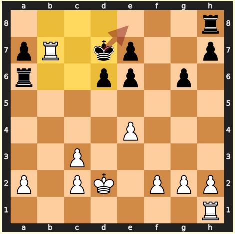


Result

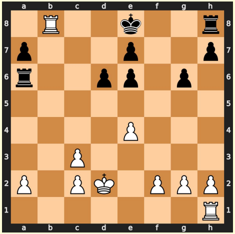


Best Move

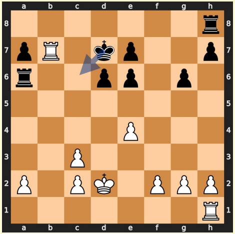
Blunders!



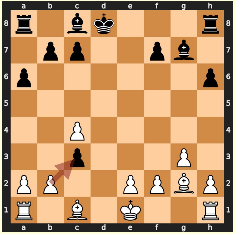
Played Move



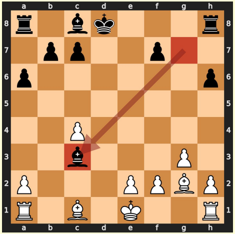
Result



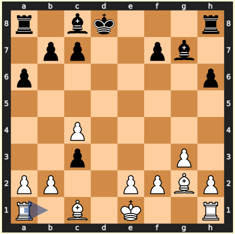
Best Move



Played Move

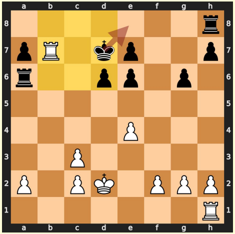


Result

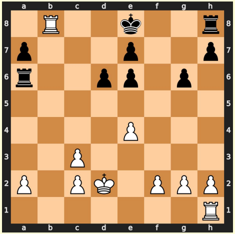


Best Move

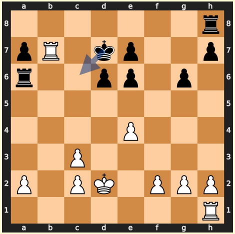
Blunders!



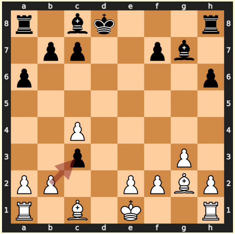
Played Move



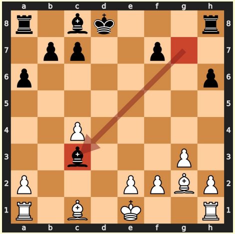
Result



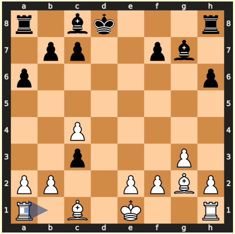
Best Move



Played Move



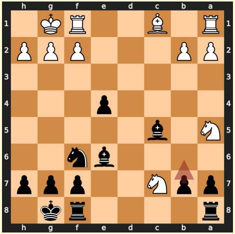
Result



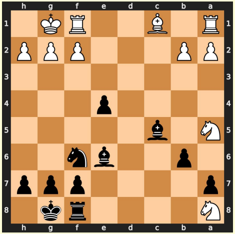
Best Move

Our LC0 evaluation misled us in these cases. But why didn't it in 2021?

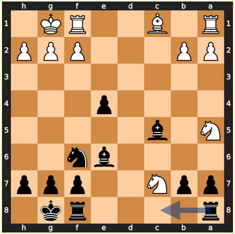
Blunders Happened in 2021, Too!



Played Move

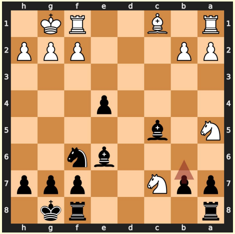


Result

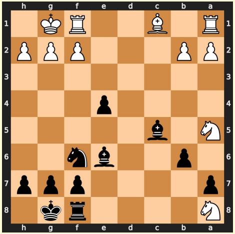


Best Move

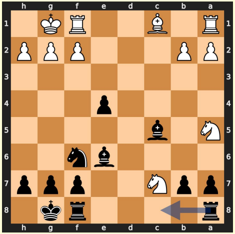
Blunders Happened in 2021, Too!



Played Move



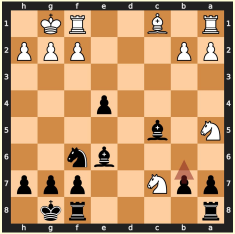
Result



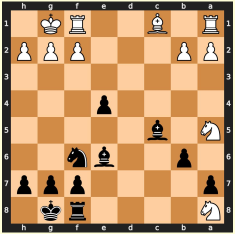
Best Move

We had not paid attention to the 2021 blunders because . . .

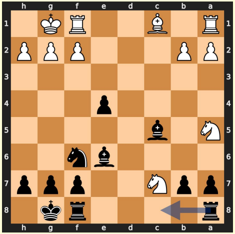
Blunders Happened in 2021, Too!



Played Move



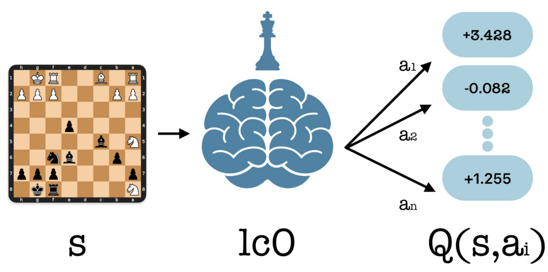
Result

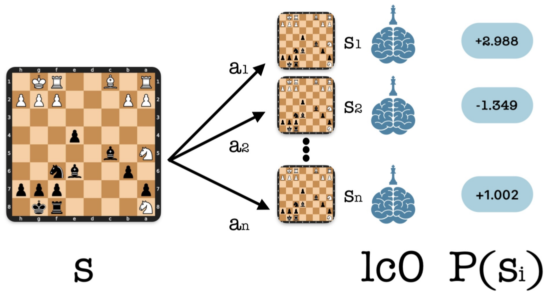


Best Move

We had not paid attention to the 2021 blunders because . . . we won anyway!

Fix: Next-state Evaluation for Top Few Actions





Fianchetto Updated vs. StrangeFish2

Agents	Win	Draw	Loss
Fianchetto (old) vs. StrangeFish2	248	25	227
Fianchetto (updated) vs. StrangeFish2	300	28	172

Table: Performance of (2022) competition version and updated version of Fianchetto against StrangeFish2 (released after 2022 competition) over 500 games.

Outline

1. Challenges of RBC
2. Baseline agent: StrangeFish
3. Our agent: Fianchetto
4. Competition results and analysis
5. Conclusion

Summary and Outlook

- RBC a new, exciting prospect for research on imperfect information games. Almost no public information, long horizon, constraints on compute time, ...

Summary and Outlook

- RBC a new, exciting prospect for research on imperfect information games. Almost no public information, long horizon, constraints on compute time, ...
- Fianchetto a thoughtfully-engineered agent with sound basis, but subject to questionable assumptions!

30 / 31

Summary and Outlook

- RBC a new, exciting prospect for research on imperfect information games. Almost no public information, long horizon, constraints on compute time, ...
- Fianchetto a thoughtfully-engineered agent with sound basis, but subject to questionable assumptions!

- How to best transfer knowledge from Chess to RBC?
- MCTS—workhorse of modern game-playing—doubly confounded by hidden state and compute time in RBC.
- How to transfer the successes of deep learning on sequential data (speech, NLP) to RBC?
- How to gather lots of useful training data for RBC?
- Benchmark RBC against humans.

Summary and Outlook

- RBC a new, exciting prospect for research on imperfect information games. Almost no public information, long horizon, constraints on compute time, ...
- Fianchetto a thoughtfully-engineered agent with sound basis, but subject to questionable assumptions!

- How to best transfer knowledge from Chess to RBC?
- MCTS—workhorse of modern game-playing—doubly confounded by hidden state and compute time in RBC.
- How to transfer the successes of deep learning on sequential data (speech, NLP) to RBC?
- How to gather lots of useful training data for RBC?
- Benchmark RBC against humans.

- Lots of science waiting to be done on RBC!

Summary and Outlook

- RBC a new, exciting prospect for research on imperfect information games. Almost no public information, long horizon, constraints on compute time, ...
- Fianchetto a thoughtfully-engineered agent with sound basis, but subject to questionable assumptions!

- How to best transfer knowledge from Chess to RBC?
- MCTS—workhorse of modern game-playing—doubly confounded by hidden state and compute time in RBC.
- How to transfer the successes of deep learning on sequential data (speech, NLP) to RBC?
- How to gather lots of useful training data for RBC?
- Benchmark RBC against humans.

- Lots of science waiting to be done on RBC!

Thank you!

References

PL21 Gian-Carlo Pascutto and Gary Linscott. Leela Chess Zero. URL: https://lczero.org. 2021.

RCK21 Tord Romstad, Marco Costalba, and Joona Kiiski. Stockfish–Open Source Chess Engine. URL: https://stockfishchess.org. 2021.

T94 Gerald Tesauro. TD-Gammon, a Self-Teaching Backgammon Program, Achieves Master-Level Play. *Neural Computation* 6(2): 215–219, 1994.

CHH02 Murray Campbell, A. Joseph Hoane Jr., Feng-hsiung Hsu. Deep Blue. *Artificial Intelligence* 134 (1–2): 57–83, 2002.

S+16 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.

S02 Brian Sheppard. World-championship-caliber Scrabble. *Artificial Intelligence* 134 (1–2): 241–275, 2002.

BS19 Noam Brown and Tuomas Sandholm. Superhuman AI for multiplayer poker. Science 365 (6456): 885–890. 2019.

M+17 Matej Moravcík, Martin Schmid, Neil Burch, Viliam Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael H. Bowling. DeepStack: Expert-level artificial intelligence in heads-up no-limit poker. *Science* 356 (6337): 508–513, 2017.

MGL18 Jared Markowitz, Ryan W. Gardner, and Ashley J. Llorens. On the complexity of Reconnaissance Blind Chess. *CoRR*, abs/1811.03119, 2018.

P+22 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-Bapriste 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.