Elliot Harris

May 25, 2022

Sven Anderson

Intro to AI

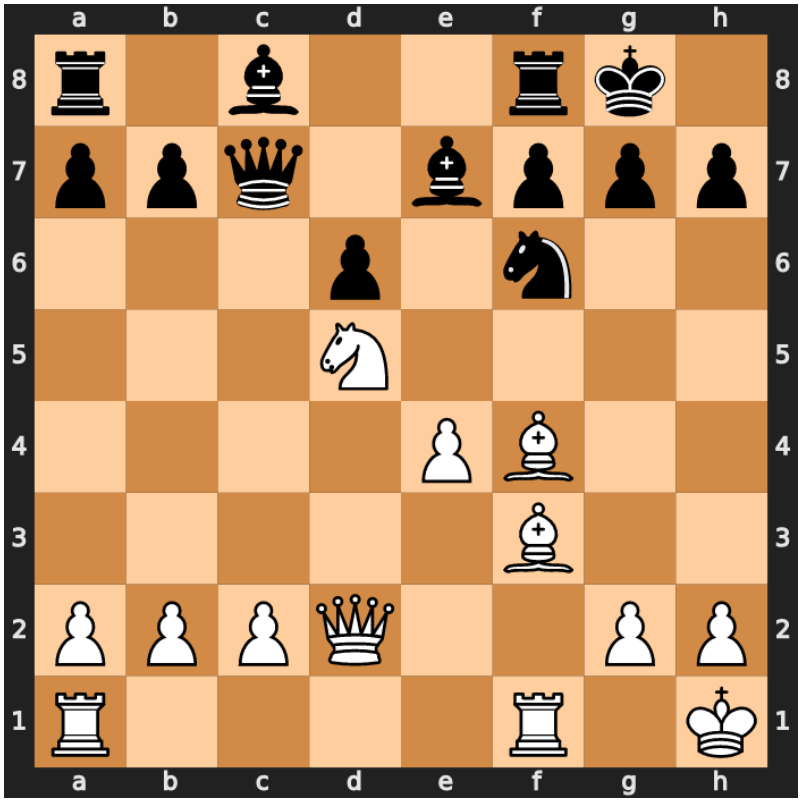
Teaching a Computer to Play Chess

The modern game of Chess spans back thousands of years, and is thought to have originally been created in India. Since then, humans have become increasingly better at playing, to the extent that it is now a worldwide sport with bi-annual world championships for millions of dollars. Much more recently however, with the advent of computers, humans have passed on their talents to the likes of binary machines. What may seem like a trivial problem amidst a world of much more pressing issues than what moves to play next, the relative strength of computers in chess actually represents something far greater than game-playing. As computers get better, so do their chess playing skills. Today, an undergrad programmer like myself can create an engine that can beat most novice chess players. While I will delve deeper into the how, a brief overview of my methods should be identified. I used a minimax algorithm, as well as alpha-beta pruning to enhance said algorithm. I coded a homemade evaluation function in order to evaluate positions, and I implemented iterative deepening in order to limit my search tree based on time.

Now I will dive into the specifics of how I programmed my engine. Using Python, I utilized a library called Python chess, which handles move-validation, generation and all the nitty gritty rules of chess. I used another library called Flask which allowed me to host a local server and run my engine so I can quickly test and visualize its outputs. Much of this was

recommended in a Medium article I have linked in the bibliography. Once I was able to run the engine, see it play random moves, I knew I could get to work building it.

The first and most important aspect of the engine is the evaluation function. Without a way to evaluate, we can't search for anything. My evaluation function has 5 factors. First, it looks at material. I define material as the number of pawns a certain piece is worth. So a knight for example, is worth 3.1 pawns, while a Rook is 4.5, etc. The evaluation calculates which side has more material, with the initial material being equal of course. The next feature is what stage of the game we are in. I defined four stages of the game based on how many pieces that were not pawns were left. The first phase is called the middlegame, which is a stage that covers the majority of moves for the majority of games. Any position in which the cumulative amount of non-pawns is greater than 4, or 5 if there are queens, counts as a middlegame position in my evaluation function. If there are less than that, but still something, that counts as the endgame, and if there are no pisces left for a side, the phase changes to corner king. The value of having different phases is to weigh the various evaluation metrics. For example, the piece tables of where a piece is most powerful becomes irrelevant in the corner king phase, as we are only focused on checkmating the opposing king.



In this position, it evaluates white as winning by 65 percent of a pawn, due to the isolated pawn on the d-file for black, and whites central Knight.

The other aspect of the engine, once it knows how to evaluate, is to search. The engine uses a lot of cool search heuristics and functions to try and speed up the search. Because python is such a slow language, in my trials of running a simple min-max algorithm, it can be reasonable at about depth 3. Depth 4 or greater takes at least a few minutes depending on the position. Luckily, I was able to implement alpha beta pruning, which takes what used to be a few minutes at depth 4 and can often result in less than a minute. Other heuristics, such as a transposition table, or move ordering were used to speed up the process. I won't get into the nitty gritty of everything used because I only have 1,000 words.

My methods to test the engine were simple. I developed 10 ‘test’ positions, where I knew of the correct move. These positions ranged from a checkmate in 4, to a complex endgame. I would test the amount of nodes expanded, the time taken to find the right move, or if a move was found at all. Some of these the engine was able to find only if given multiple minutes to search. Once I felt satisfied with the results, I played three different opponents to gauge the engine's playing strength. First, I played a chess.com beginner bot, estimated to be rated 850 elo. My engine wiped the floor, winning in 16 moves, and according to chess.com’s result analysis, playing with no mistakes or blunders. I will put all the games in pgn format at the end of this report. Next, I played by myself, giving the engine 10 secs of searching plus any additional time at the depth it ended at in its iterative deepening function. I did not record the games, but the result was it beat me once, and lost once. However, qualitatively, the game the engine one was because of a bad blunder on my end, and the game it lost was mostly a result of small mistakes by the engine over time. Needing an opponent that resided in between the ratings of a beginner bot and my 2100 rapid rating, I had it play my friend who is about 1400 elo. Those three games are linked below. It is interesting to note that the engine was able to win with almost perfect accuracy the first game, and kept up on time for a game with 15 minutes and 10 second increment for each side. A bug was found, in which the engine seemingly prefers to stalemate in positions where it also has checkmate. As such, my friend was able to sneak out with two draws in positions he had no hope of winning. Here are some estimates of my engine's performance in terms of how efficiently it searches.

Time per Search Given	Depth reached	Nodes Searched
3 seconds + an average of about 30 seconds at the current depth. You can look at the recorded times in the pgns	Between 3 and 7 ply, average of 4 in the middlegame and 5 in the endgame or simpler positions (like check)	Anywhere between 8,000 and half a million, average of about 100,000

for more specific time stamps	Note: It almost never reached a depth of 8, and would get stuck at 3 in only very complicated tactical middlegames	Note: Positions in which it got stuck at a later depth and had a lot of possibilities took up most of the search.

To conclude, I will point out where my engine was successful, and where it failed. The first obvious failure is the bug where it seems to purposefully stalemate. This bug clearly reduces my engine's results as it leads to a lot of draws in positions it otherwise would win. However, this bug is actually pretty funny to me, and I rather enjoy the comedic effect of destroying an opponent only to force the game into a draw. The biggest success is its overall playing strength in the opening and middle game. I would estimate based on my experience with chess and the results against three different opponents, it plays the first two stages of the game at atleast an 1800 strength. It falters heavily in the endgame, when depths of 5 or 6 don't suffice in most positions. The two biggest factors that seem to improve its strength are the move ordering and the alpha beta pruning. These two features alone make the search efficient enough to be an intelligent and interesting player. In the future, there are a couple of things I would add or change to make it even better. First, I would obviously fix the stalemate bug. Secondly, I would try to pass more information between the iterations in its iterative deepening function. Right now, it only knows what the last iteration found as the best move. Passing in multiple candidate moves would improve the search time tremendously. Thirdly, I would improve the quiet confirmation function. It uses almost no pruning or heuristics outside of the base alpha beta pruning. It also doesn't look at checks, so it can often completely miss-evaluate a position. Lastly, I would improve the evaluation function, especially in the endgame. It needs much more testing and complexity to reach the same level of accuracy it achieves in the middlegame.

Bibliopgraphy

Chessprogramming Wiki, https://www.chessprogramming.org/Main_Page.

Gaikwad, Ansh. “Let's Create a Chess AI.” *Medium*, Developer Students' Community Vishwakarma Institute of Technology, Pune, 14 Oct. 2020, https://medium.com/dscvitpune/lets-create-a-chess-ai-8542a12afef.

Team, Chess.com. “History of Chess: From Early Stages to Magnus.” *Chess.com*, Chess.com, 30 Sept. 2019, https://www.chess.com/article/view/history-of-chess#:~:text=Chess%2C%20as%20we%20know%20it,Spanish%20priest%20named%20Ruy%20Lopez.

PGN's of games played:

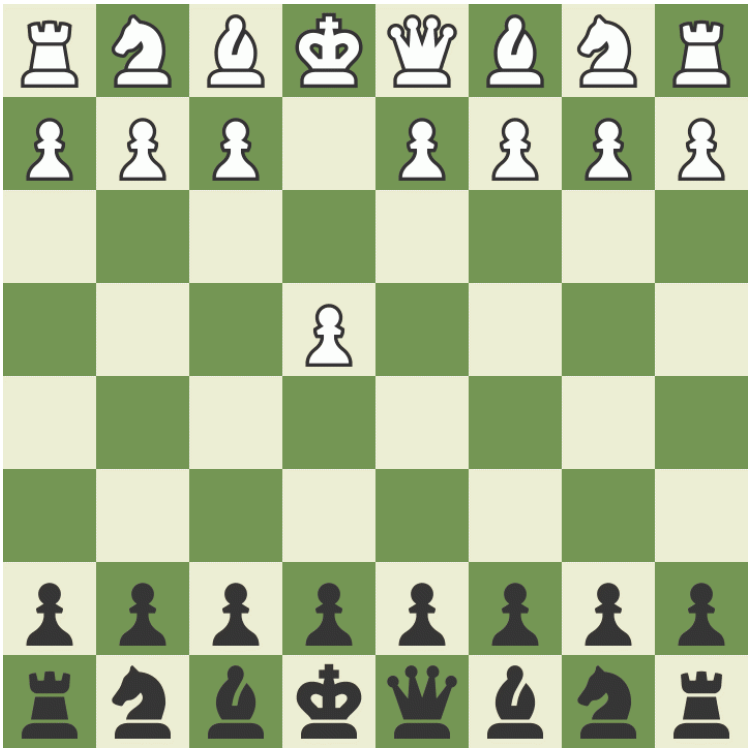
My computer played a match against chess.com bot named Karim (elo of 850), here is the gif

1. e4 f5 2. exf5 Nf6 3. d4 Nc6 4. d5 Kf7 5. dxc6 e6 6. cxd7 Bxd7 7. Bc4 Qe7 8. Qf3 Bc6 9. Bxe6+ Ke8 10. Qh3 h5 11. Ne2 Rd8 12. Nbc3 Qb4 13. Qg3 Ng4 14. Kf1 Bc5 15. Qxc7 Bxf2 16. Qf7# 1-0

Game 1 vs Finn (1400 elo)

1. e4 {[%clk 0:15:10]} 1... c6 {[%clk 0:15:10]} 2. c4 {[%clk 0:15:14.6]} 2... d5 {[%clk 0:15:14.1]} 3. exd5 {[%clk 0:15:18.5]} 3... cxd5 {[%clk 0:15:21.1]} 4. cxd5 {[%clk 0:15:18.4]} 4... Nf6 {[%clk 0:15:14.7]} 5. Nc3 {[%clk 0:15:17.6]} 5... Nxd5 {[%clk 0:15:13]} 6. Nf3 {[%clk 0:15:16.3]} 6... Nxc3 {[%clk 0:15:03.3]} 7. bxc3 {[%clk 0:15:16.7]} 7... Nc6 {[%clk 0:15:09.1]} 8. d4 {[%clk 0:15:14.4]} 8... Bg4 {[%clk 0:15:11.2]} 9. d5 {[%clk 0:14:20.9]} 9... Ne5 {[%clk 0:15:01.6]} 10. Nxe5 {[%clk 0:13:54.9]} 10... Bxd1 {[%clk 0:14:14.3]} 11. Bb5+ {[%clk 0:13:46.5]} 11... Qd7 {[%clk 0:14:21.8]} 12. Bxd7+ {[%clk 0:13:36]} 12... Kd8 {[%clk 0:14:28.4]} 13. Kxd1 {[%clk 0:13:23.6]} 13... Kc7 {[%clk 0:14:34]} 14. Nxf7 {[%clk 0:13:10.2]} 14... Rg8 {[%clk 0:14:35.5]} 15. Bf5 {[%clk

0:12:54.4]} 15... g6 {[%clk 0:14:37.4]} 16. Be6 {[%clk 0:12:36.2]} 16... Bg7 {[%clk 0:14:26]} 17. Bg5 {[%clk 0:12:11.7]} 17... Rge8 {[%clk 0:14:01.8]} 18. Bxe7 {[%clk 0:11:41.4]} 18... b6 {[%clk 0:13:27.5]} 19. Kc2 {[%clk 0:10:54.3]} 19... Kb7 {[%clk 0:13:30.8]} 20. Nd6+ {[%clk 0:10:28]} 20... Ka6 {[%clk 0:13:31.8]} 21. Nxe8 {[%clk 0:10:08.4]} 21... Rxe8 {[%clk 0:13:40.5]} 22. Bg5 {[%clk 0:10:02.9]} 22... h6 {[%clk 0:13:41.7]} 23. Be3 {[%clk 0:09:59.3]} 23... Kb7 {[%clk 0:13:44.7]} 24. Rhg1 {[%clk 0:09:47.3]} 24... Kc7 {[%clk 0:13:48.3]} 25. Rad1 {[%clk 0:09:40.7]} 25... Kd6 {[%clk 0:13:50.4]} 26. Bf7 {[%clk 0:09:38.6]} 26... Rc8 {[%clk 0:13:52.6]} 27. Bf4+ {[%clk 0:09:37.5]} 27... Ke7 {[%clk 0:13:56.2]} 28. Bxg6 {[%clk 0:09:24.9]} 28... Rxc3+ {[%clk 0:14:01.4]} 29. Kb1 {[%clk 0:09:20.2]} 29... a5 {[%clk 0:13:51.5]} 30. Rde1+ {[%clk 0:09:14.5]} 30... Kd7 {[%clk 0:13:38]} 31. Bf5+ {[%clk 0:09:08.9]} 31... Kd8 {[%clk 0:13:45.9]} 32. Re6 {[%clk 0:09:05.2]} 32... b5 {[%clk 0:13:45.9]} 33. Rge1 {[%clk 0:09:04.5]} 33... Kc8 {[%clk 0:13:30.6]} 34. Re7+ {[%clk 0:08:49.6]} 34... Kd8 {[%clk 0:13:36.8]} 35. Re8# {[%clk 0:08:35.2]} 1-0

Game 2 vs Finn 

1. e4 {[%clk 0:15:10]} 1... c5 {[%clk 0:15:10]} 2. d4 {[%clk 0:15:17]} 2... cxd4 {[%clk 0:15:03.8]} 3. c3 {[%clk 0:15:25]} 3... dxc3 {[%clk 0:15:08.4]} 4. Nxc3 {[%clk 0:15:32.6]} 4... Nc6 {[%clk 0:15:12]} 5. Nf3 {[%clk 0:15:40.7]} 5... e6 {[%clk 0:15:15.1]} 6. Bc4 {[%clk 0:15:49.1]} 6... a6 {[%clk 0:15:17.7]} 7. O-O {[%clk 0:15:57]} 7... b5 {[%clk 0:15:18.3]} 8. Bb3 {[%clk 0:16:04.6]} 8... Bb7 {[%clk 0:15:22.1]} 9. Bg5 {[%clk 0:15:59.1]} 9... Be7 {[%clk 0:15:00.7]} 10. Bxe7 {[%clk 0:15:40.6]} 10... Kxe7 {[%clk 0:14:38.3]} 11. Re1 {[%clk 0:15:40.5]} 11... b4 {[%clk 0:14:31.4]} 12. Nd5+ {[%clk 0:15:19.7]} 12... exd5 {[%clk 0:14:22.3]} 13. Qxd5 {[%clk 0:15:25.6]} 13... Qf8 {[%clk 0:14:12.5]} 14. Rad1 {[%clk 0:14:44.8]} 14... Nf6 {[%clk 0:13:40.7]} 15. Qc5+ {[%clk 0:14:23.1]} 15... Ke8 {[%clk 0:12:54.3]} 16. Qxf8+ {[%clk 0:13:33.7]} 16... Rxf8 {[%clk 0:12:34.6]} 17. e5 {[%clk 0:13:39.9]} 17... Ng4 {[%clk 0:12:33.1]} 18. h3 {[%clk 0:13:16.3]} 18... Nh6 {[%clk 0:12:22]} 19. e6 {[%clk 0:12:16.7]} 19... fxe6

{[%clk 0:12:17.8]} 20. Ng5 {[%clk 0:12:06.2]} 20... Nf5 {[%clk 0:12:11.2]} 21. Nxh7 {[%clk 0:11:12.2]} 21... Rf7 {[%clk 0:11:25.8]} 22. Ng5 {[%clk 0:11:08.7]} 22... Re7 {[%clk 0:11:21.6]} 23. Nf3 {[%clk 0:10:39]} 23... d5 {[%clk 0:11:09.8]} 24. g4 {[%clk 0:10:30.2]} 24... Nd6 {[%clk 0:11:03.1]} 25. Ng5 {[%clk 0:10:31.1]} 25... Kd7 {[%clk 0:11:01.5]} 26. Ba4 {[%clk 0:10:27.7]} 26... Rae8 {[%clk 0:10:58.1]} 27. Nf3 {[%clk 0:10:31.6]} 27... Nc4 {[%clk 0:10:54.3]} 28. Ne5+ {[%clk 0:10:10.6]} 28... Nxe5 {[%clk 0:10:49.7]} 29. Rxe5 {[%clk 0:10:18.8]} 29... Rd8 {[%clk 0:10:43.4]} 30. Rde1 {[%clk 0:10:17.2]} 30... Kd6 {[%clk 0:10:32.5]} 31. Bxc6 {[%clk 0:10:11.6]} 31... Bxc6 {[%clk 0:10:27.4]} 32. f4 {[%clk 0:10:09.7]} 32... Rde8 {[%clk 0:10:15.9]} 33. Kf2 {[%clk 0:10:07.8]} 33... Kd7 {[%clk 0:10:00.7]} 34. h4 {[%clk 0:10:14.1]} 34... Bb7 {[%clk 0:09:52]} 35. g5 {[%clk 0:10:18]} 35... Rf7 {[%clk 0:09:39.2]} 36. Kf3 {[%clk 0:10:20.4]} 36... Rff8 {[%clk 0:09:24.4]} 37. h5 {[%clk 0:10:16.5]} 37... Rh8 {[%clk 0:09:13.1]} 38. Kg4 {[%clk 0:10:03.2]} 38... Bc6 {[%clk 0:09:09.2]} 39. Rh1 {[%clk 0:10:07]} 39... Kc7 {[%clk 0:08:45.5]} 40. h6 {[%clk 0:10:08.8]} 40... gxh6 {[%clk 0:08:34.3]} 41. Rxh6 {[%clk 0:10:01.8]} 41... Rxh6 {[%clk 0:08:27.7]} 42. gxh6 {[%clk 0:10:09.4]} 42... Rg8+ {[%clk 0:08:19.7]} 43. Rg5 {[%clk 0:10:10.9]} 43... Rh8 {[%clk 0:08:11.7]} 44. Rg7+ {[%clk 0:09:42.9]} 44... Bd7 {[%clk 0:08:04.4]} 45. Kg5 {[%clk 0:09:42.4]} 45... d4 {[%clk 0:07:52.4]} 46. h7 {[%clk 0:09:18.3]} 46... d3 {[%clk 0:07:44.8]} 47. Kh6 {[%clk 0:09:25.1]} 47... d2 {[%clk 0:07:42.7]} 48. Rg8 {[%clk 0:09:32.8]} 48... Rxh7+ {[%clk 0:07:29.6]} 49. Kxh7 {[%clk 0:09:40.2]} 49... d1=Q {[%clk 0:07:24.3]} 50. Ra8 {[%clk 0:09:40.5]} 50... Qh1+ {[%clk 0:07:12.5]} 51. Kg7 {[%clk 0:09:44.8]}

51... Qxa8 {[%clk 0:07:09.7]} 52. Kf6 {[%clk 0:09:52.8]} 52... Qh8+ {[%clk 0:07:06.8]} 53. Kg5 {[%clk 0:09:59.7]} 53... Qxb2 {[%clk 0:06:59.1]} 54. f5 {[%clk 0:10:06.4]} 54... exf5 {[%clk 0:06:56.6]} 55. a4 {[%clk 0:10:13.8]} 55... bxa3 {[%clk 0:06:45]} 56. Kg6 {[%clk 0:10:18.6]} 56... Qh8 {[%clk 0:06:39.8]} 57. Kg5 {[%clk 0:10:24.8]} 57... Qg8+ {[%clk 0:06:35]} 58. Kf4 {[%clk 0:10:33.1]} 58... Qg2 {[%clk 0:06:29.6]} 59. Ke3 {[%clk 0:10:38.8]} 59... a2 {[%clk 0:06:20.7]} 60. Kd3 {[%clk 0:10:38.3]} 60... a1=Q {[%clk 0:06:10.7]} 61. Ke3 {[%clk 0:10:44]} 61... Qe4+ {[%clk 0:05:53.8]} 62. Kd2 {[%clk 0:10:51.6]} 62... Be8 {[%clk 0:05:34.5]} 1/2-1/2

Game 3 vs Finn

1. d4 {[%clk 0:15:10]} 1... d5 {[%clk 0:15:10]} 2. c4 {[%clk 0:15:13.9]} 2... Nc6 {[%clk 0:15:17.2]} 3. Nc3 {[%clk 0:15:16.7]} 3... Nf6 {[%clk 0:15:23.6]} 4. Nf3 {[%clk 0:15:20.2]} 4... Bf5 {[%clk 0:15:26.9]} 5. Bf4 {[%clk 0:15:22.8]} 5... a6 {[%clk 0:15:30.6]} 6. cxd5 {[%clk 0:15:12.4]} 6... Nxd5 {[%clk 0:15:34.9]} 7. Nxd5 {[%clk 0:15:10.7]} 7... Qxd5 {[%clk 0:15:41.8]} 8. Bxc7 {[%clk 0:14:05.4]} 8... e6 {[%clk 0:15:35]} 9. e3 {[%clk 0:13:53.6]} 9... Bb4+

{[%clk 0:15:33.7]} 10. Nd2 {[%clk 0:13:46.5]} 10... Rc8 {[%clk 0:15:20.3]} 11. Bg3 {[%clk 0:13:31.7]} 11... Nxd4 {[%clk 0:14:22.3]} 12. exd4 {[%clk 0:13:30.9]} 12... Qxd4 {[%clk 0:14:23.6]} 13. a3 {[%clk 0:13:26.5]} 13... Bxd2+ {[%clk 0:13:07]} 14. Qxd2 {[%clk 0:13:14.3]} 14... Qxd2+ {[%clk 0:13:14.3]} 15. Kxd2 {[%clk 0:13:10.4]} 15... Rc2+ {[%clk 0:13:22.2]} 16. Ke1 {[%clk 0:13:01.4]} 16... O-O {[%clk 0:13:28.6]} 17. b4 {[%clk 0:12:38.2]} 17... Rfc8 {[%clk 0:13:14.3]} 18. Be2 {[%clk 0:12:26.3]} 18... Rc1+ {[%clk 0:13:08.5]} 19. Rxc1 {[%clk 0:12:21.1]} 19... Rxc1+ {[%clk 0:13:15.8]} 20. Bd1 {[%clk 0:12:15.2]} 20... Ra1 {[%clk 0:13:09.4]} 21. b5 {[%clk 0:12:05.7]} 21... Rxa3 {[%clk 0:13:05.5]} 22. b6 {[%clk 0:12:00.1]} 22... Ra1 {[%clk 0:13:00.3]} 23. Be5 {[%clk 0:11:45.2]} 23... Rb1 {[%clk 0:13:01.4]} 24. Bd4 {[%clk 0:11:37.6]} 24... f6 {[%clk 0:13:01.2]} 25. Be3 {[%clk 0:11:07.9]} 25... a5 {[%clk 0:13:07.5]} 26. Bc5 {[%clk 0:11:00.8]} 26... e5 {[%clk 0:12:39.5]} 27. Kd2 {[%clk 0:10:34.5]} 27... Rb2+ {[%clk 0:12:07.6]} 28. Kc3 {[%clk 0:10:28.4]} 28... Rb1 {[%clk 0:11:59.9]} 29. Bb3+ {[%clk 0:10:18.9]} 29... Kh8 {[%clk 0:11:54.4]} 30. Rxb1 {[%clk 0:10:16.1]} 30... Bxb1 {[%clk 0:12:02.5]} 31. Bd5 {[%clk 0:10:12.8]} 31... a4 {[%clk 0:11:58.9]} 32. Bxb7 {[%clk 0:10:11.2]} 32... Bf5 {[%clk 0:12:05]} 33. Bc6 {[%clk 0:09:54.8]} 33... Bc8 {[%clk 0:12:13.1]} 34. Kb4 {[%clk 0:09:43.2]} 34... Kg8 {[%clk 0:12:13.8]} 35. Kxa4 {[%clk 0:09:30.7]} 35... Kf7 {[%clk 0:12:21]} 36. Bd5+ {[%clk 0:09:24.6]} 36... Kg6 {[%clk 0:12:28.8]} 37. Ka3 {[%clk 0:09:07.3]} 37... Kf5 {[%clk 0:12:36.4]} 38. b7 {[%clk 0:08:55.7]} 38... Bxb7 {[%clk 0:12:44.1]} 39. Bxb7 {[%clk 0:08:47.4]} 39... h5 {[%clk 0:12:51.7]} 40. Be3 {[%clk 0:08:37.1]} 40... g5 {[%clk 0:12:45]} 41. Ka2 {[%clk

0:08:26.8]} 41... e4 {[%clk 0:12:38.5]} 42. Bd4 {[%clk 0:07:58]} 42... g4 {[%clk 0:12:30.1]} 43. g3 {[%clk 0:07:47.6]} 43... Kg5 {[%clk 0:12:03.9]} 44. Bxe4 {[%clk 0:07:42.9]} 44... f5 {[%clk 0:12:12]} 45. h4+ {[%clk 0:07:41.2]} 45... Kg6 {[%clk 0:12:15.7]} 46. Bd3 {[%clk 0:07:33.9]} 46... Kf7 {[%clk 0:12:22.2]} 47. Bxf5 {[%clk 0:07:28.3]} 47... Ke7 {[%clk 0:12:29.7]} 48. Bg6 {[%clk 0:07:16.6]} 48... Kd8 {[%clk 0:12:39.6]} 49. Bxh5 {[%clk 0:07:15.2]} 49... Kc7 {[%clk 0:12:49.5]} 50. Bxg4 {[%clk 0:07:09.9]} 50... Kb8 {[%clk 0:12:59.4]} 51. Be5+ {[%clk 0:06:56.9]} 51... Ka8 {[%clk 0:13:09.3]} 52. Bc8 {[%clk 0:06:48.7]} 52... Ka7 {[%clk 0:13:01.6]} 53. Bc7 {[%clk 0:06:43.9]} 53... Ka8 {[%clk 0:13:09.1]} 54. Kb3 {[%clk 0:06:40.8]} 54... Ka7 {[%clk 0:13:16.6]} 55. Kc4 {[%clk 0:06:39.2]} 55... Ka8 {[%clk 0:13:24.4]} 56. Ba6 {[%clk 0:06:39.3]} 56... Ka7 {[%clk 0:13:32.3]} 57. Kb5 {[%clk 0:06:20.8]} 57... Ka8 {[%clk 0:13:40.9]} 58. Kb6 {[%clk 0:06:02.9]} 1/2-½