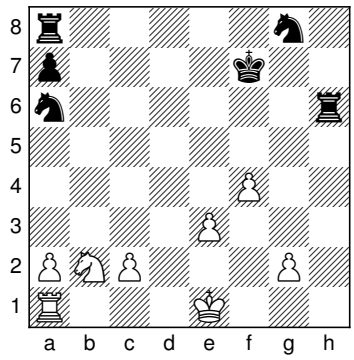
Losing Chess: 1. e3 wins for White

Mark Watkins[1]

University of Sydney, School of Mathematics and Statistics

We provide an update to our previous news item (Watkins, 2014) concerning the game of Losing Chess. At that time, we had solved 18 of the 20 Black responses to 1. e3 as being won for White, with only b6 and c5 remaining. We are pleased to report that we have now completed the proof that 1. e3 wins against any Black defense. A fuller description of our work appears in the (final) status report on the main project website (Watkins, 2016). Currently a browseable version of the final proof is also available. Herein we provide some brief information.

As indicated in the previous report, before solving 1. e3 e6 our code went through a major rewrite, involving making our software "cluster-capable" (we typically used around 128 cores) and building various 6-unit tablebases, such as those where Black had a lone King and White had no Queens or Bishops. The "automated" part of the solving of 1. e3 c5 went pretty well with the large hardware, and indeed within a couple of months we had solved all but one line (1. e3 c5 2. Bb5 Qc7), including a quite lengthy proof with 2... c4. However, in this last line our heuristics did not work so well, and we followed more false leads. In the end, we built some relevant tablebases, here 4-vs-KK and 4-vs-KN, which allowed us to see from afar whether our approach was workable. The crucial position can be seen in Figure 1 (left), where it seemed for some time that White should win by 19. Nd1, and only after much searching was a refutation found. Contrarily, not wanting to waste the work we had done in these lines leading to White's 19th move, we spent longer than perhaps normal on various moves with lower scores, and with the aid of the above tablebases found (somewhat unexpectedly) that 19. g3 eventually leads to a win (finished in Feb 2015). The final proof size is 217046510 nodes, of which 2... c4 is just under half, and 2... Nh6 is slightly larger than 2... Qc7 (36741859 nodes), even though the latter is by far the most difficult to discover.



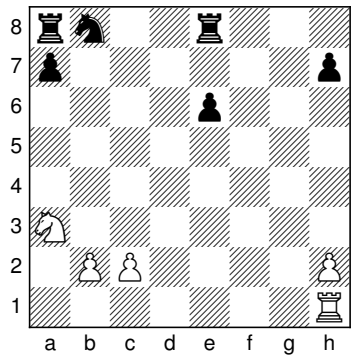


Figure 1: at left, a key position in solving 1. e3 c5 where 19. Nd1 seems to fail but 19. g3 succeeds (quite contrary to our heuristics). At right, a position from 1. e3 b6 which needs some searching to see that 17. c4 wins for White (again difficult heuristically).

A similar storyline emerged in our 1. e3 b6 proof. Firstly we had strong hopes for solving this already in 2013 via 2. Ba6, but that move never seemed to work out. We switched to the relatively unexplored 2. a4 in May 2015, and after a couple of months had done significant work in solving 2... b5 (over 137 million nodes), and also 2... e6 3. Ra3 Bxa3 4. Nxa3 b5 (over 243 million nodes). However, we still had 4... Qh4 to solve in this last line. As it transpired, we wasted the next 14 months looking at 5. h3, as unfortunately our heuristics (either automated or with human aid) did not prod us enough to look at 5. a5 bxa5 6. Qh5!, when after a race with both sides making various captures with their Queen, an endgame is reached which needs a bit searching to become recognizable as being hopeful for White to win (see Figure 1, right). This is similar to the situation above with 19. g3 in the 1. e3 c5 line, but the resulting endgame here actually resolves itself more easily (after 17. c4). Again the final proof size (2.6 million nodes) is paltry compared to the difficulty of discovering it in the first place. The final proof for 1. e3 b6 takes 448616089 nodes, or just over half of the size of the total proof for all 20 Black responses.

The total size of the proof is 863301867 nodes and 66187848 transposition pointers. We stored it in a compact form (6 bytes per node) and using generic compression (bzip2) reduces the size to 1.3 gigabytes. As mentioned

[1] email: watkins@maths.usyd.edu.au

in our previous article, we deliberately chose to target the solution for 4-unit tablebases, which themselves take up only 400MB. There are 804549638 internal nodes in the proof tree (actually a graph), and of the 58752229 terminal nodes, 23448460 are in tablebases, in 18117361 of them White has lost everything, and for the remaining 17186408 White is stalemated with at least one unit left. There are 79840511 positions with 5 units remaining and 90352974 with 6, indicating that we could cut about 20% of the proof tree if we chose to make such tablebases be "standard" in our proof. Approximately 87% of the internal nodes with black-to-move have only one legal move, which is perhaps typical for a game with long sequences of forced captures.

Although I have not undertaken it much myself, Klaas Steenhuis has used the software to investigate a number of other opening moves for White. Here d3, d4, and e4 are anciently known to be losses, while Nc3, Nf3, f4, and h4 were all essentially determined to be lost even before computers got involved. Steenhuis first recalculated proofs for 1. h3 and 1. b4 that were known in 2002 and 2003, and since has shown that 1. c3, 1. f3, and 1. a3 all lose, the last being by far the most complicated (180 million nodes in the final proof).

As noted in our previous communication, we cannot be said to have approached the solving of Losing Chess in a particularly scientific manner (rather as a hobby); however, now that a proof is completed, we possess a baseline for comparison of methods. For instance, in our version of PN^2-search we did not implement something as basic as the killer heuristic at either pn-nodes or upper-level nodes. There is thus some exploration possible in trying to improve our heuristics on best-first expansion, and the effect on solving time or final proof size could be analyzed. The question of whether the proof discovery process can be "fully automated" is still open; indeed, the final line for 1. e3 b6 would hopefully be found much faster with better automated heuristics, but these of course need to be balanced in the context of the entire proof.

One surprise to me in the aftermath of announcing the result is how many people (including chess grandmasters who had made some study of the game) told me that they had thought that 1. e3 would only draw, particularly against c5 or b6. The extra initiative from White's first move does not immediately seem to be that valuable, but turns out to be sufficient in the end. On the other hand, my whole involvement in the project was a sort of "gamble" that White would win in the end, as else there would be little chance of completion.

From the standpoint of complexity, we might compare our work to the solving of checkers (Schaeffer, 2007; Schaeffer *et al.*, 2007), though comparing a game that is drawn to one that is won is not necessarily the most valid. Our final proof size is approximately 900 million positions, and even with much larger tablebases (full 6-unit) this would not be reduced by more than about 20%. Table 2 of (Schaeffer, 2007) indicates that about 15 million searches of 15 (alpha-beta) or 100 (Df-pn) seconds each were considered when solving checkers, with perhaps 1/3 of these ending up being used in the final proof, and each search being roughly 10^7 positions (Schaeffer *et al.*, 2007). Comparatively, we searched for approximately 200-300 core-years at around 2 million pn-nodes per second, or approximately 10^{16} total positions searched (about 100 times as many as for checkers). As a rough estimate, we did 10^8 pn-searches of size 10^8, and the end proof used maybe a half million of these (0.5%), with approximately a 1:2000 expansion between when expanding to a full proof (so that even pn-searches that yield a solved result have significant wastage). For Losing Chess this expansion ratio scales approximately as the square root of the size of the pn-search, but it is perhaps unwarranted (yet tempting) to propose that each Df-pn search in the checkers solution should have a similar expansion ratio to a full proof, particularly since (Schaeffer *et al.*, 2007) claims that it would take "many tens of terabytes" to save an entire proof tree from the opening position to 10-unit tablebases (which themselves contain $\approx 4 \cdot 10^{13}$ positions in approximately 250GB). It is our inference (or suspicion) that, similar to the disparate percentages of upper-level searches ultimately used in the final proofs, successful Df-pn searches of size 10^7 in checkers could typically have minimal trees of size 10^6 or more; and moreover we suspect this differential behavior is due to checkers being a draw and Losing Chess a win.

1. REFERENCES

J. Schaeffer, *Game Over: Black to Play and Draw in Checkers*, ICGA Journal **30/4** (2007), 187–197.

J. Schaeffer, N. Burch, Y. Björnsson, A. Kishimoto, M. Müller, R. Lake, P. Lu, S. Sutphen, *Checkers Is Solved*, Science **317/5844** (14 Sep 2007), 1518–1522.

M. Watkins, *Solved Openings in Losing Chess*, ICGA Journal **37/2** (2014), 106–110.

M. Watkins, (2016), `http://magma.maths.usyd.edu.au/~watkins/LOSING_CHESS`