Contents

2 Computer-Aided Analysis Methods

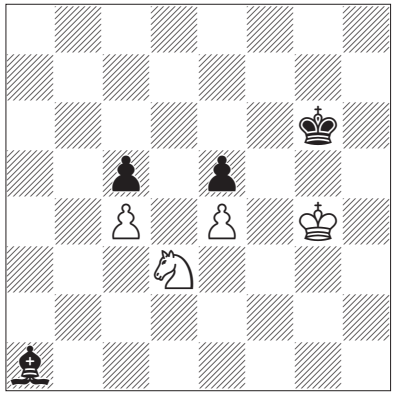
Interactive Analysis – Using a Program as a Sparring Partner

"The best sparring partner, though, is a quick, strong man who does not know anything; a madman who goes all-out, scratching, grabbing, grappling, punching, kicking, and so on."[1]

Probably the most powerful of all the possible chess analysis techniques involves using a chess program as a sparring partner, in an interactive way. What I mean by this is that the human analyst comes up with plans, and uses the computer to check the implementation of these plans. In the simplest form of interactive analysis, the computer just does blunderchecking. Perhaps the analyst will make a candidate move that a computer can quickly see drops a piece. Great! Now you don't have to spend the time (whether it be mere seconds, or perhaps minutes, or even hours) it would have taken to find the blunder without computer help. But the computer can do far more than just blunder-check plans. Computers can put up resistance... serious resistance. Often analysts are so enamoured with their own ideas, they fail to find the best way to resist them. A computer isn't enamoured with any plan, and will do what it can to defend or counter-attack.

The following position was given by Pachman in the third volume of his 1978 *Complete Chess Strategy* trilogy, to illustrate a position where programs would have great difficulty finding the right idea. Until recently all programs would have taken the c5-pawn, and even today most programs (and humans) still grab it. Pachman wrote about 1 ♘xc5 in *Complete Chess Strategy* that "White's c-pawn cannot be

W



Pachman – Hromadka
Prague Ch 1944

advanced to the queening square without the help of the king, but this allows Black to counter by attacking the e-pawn". Graham Burgess concurred in his 1997 book *The Mammoth Book of Chess* adding that "This obvious, materialistic move throws away the win!" Burgess then goes on to add "Supposing it were instead Black to move, the following variation is enlightening:

1...♗d4 2 ♘e1 ♗f2 3 ♘f3 ♔f6

Or 3...♗d4 4 ♘h4+ ♔f6 5 ♘f5.

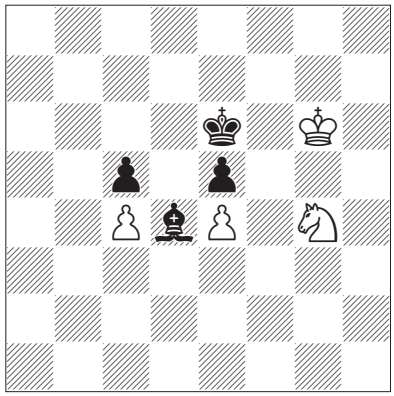
4 ♔h5 ♗g3 5 ♘h4 ♗f2 6 ♘f5 ♗g1 7 ♘h6 ♗d4 8 ♘g4+ ♔e6 9 ♔g6 *(D)*

"White will now play ♘f6-h7-g5+, etc., and win easily. However, if there were no black pawn on c5, then Black would have sufficient counterplay to hold the draw, since his king could use the c5-square to attack White's pawns. Thus in our start position, White should not take the c-pawn, but instead make progress on the kingside by manoeuvring his knight."

Burgess's argument is clear, logical and compelling. But is the position really so simple?

1 Bruce Lee, actor and martial arts master

B



Modern computers can be very effective sparring partners to help test such ideas. Here is what happens when you try the plan outlined by Burgess against a recent version of Fritz:

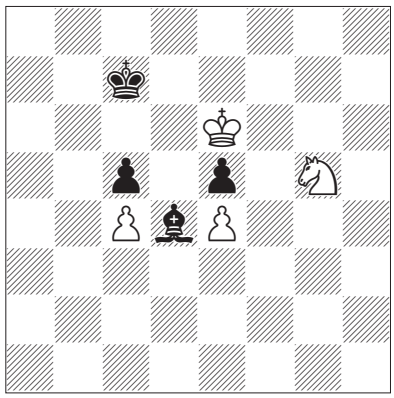
9...♗c3 10 ♘f6 ♗b2 11 ♘h7 ♔e7 12 ♔f5 ♗c3 13 ♘g5 ♗d4 14 ♘f3 ♔d6 15 ♔f6 ♔d7 16 ♘g5

16 ♘xe5+?? ♔d6 −+.

16...♔c7 17 ♔e6 *(D)*

17 ♘f7 ♔b6! 18 ♘xe5 ♔a5 19 ♔e6 ♔b4 would be a draw.

B



White can now win the e-pawn, all per the outlined plan, and Black's position looks hopeless. The knight will capture Black's e-pawn and the win is easy, right?

17...♔b6!

Black's king goes **around** the c5-square, just in time to get the counterplay White wanted to prevent by avoiding ♘xc5 in the first place.

18 ♘f7 ♔a5 19 ♔d5

If 19 ♘xe5, then 19...♔b4 =.

19...♔b4 =

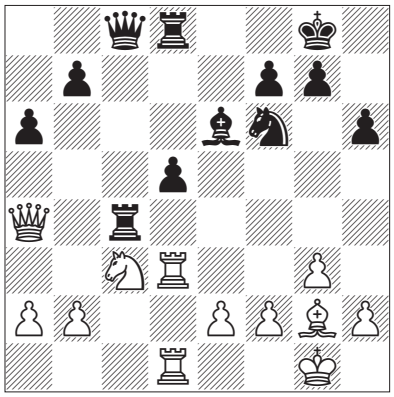
If White tries to win the e-pawn, Black's king will hover near White's c-pawn and win it as well. White has no way to make progress. Let's just double check to see what happens in a pawn race.

20 ♘xe5 ♗xe5 21 ♔xe5 ♔xc4

We have a tablebase draw.

When using a program in this interactive way, it can sometimes pay to look at the computer's lines of thinking, the 'PV'. Most programs have modes where this information can be displayed, and doing so will not slow down the search in any way. When looking at the PV that the program is displaying, don't pay too much attention to the whole line that is displayed. Some programs, because of design trade-offs made by the programmer, will often display complete nonsense in their PV. The first move of the line will be displayed perfectly correctly, and the position evaluation is fine, but later moves can truly be complete nonsense. Here is an example:

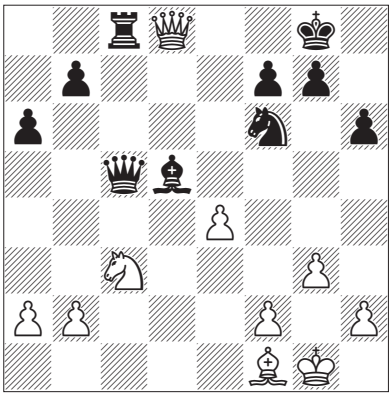
W



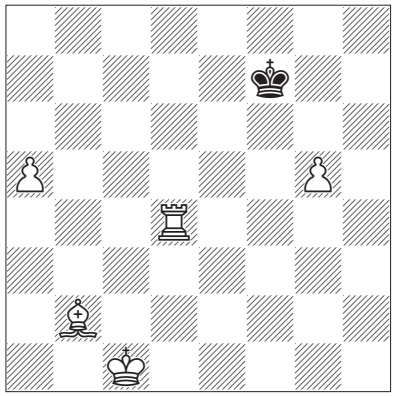
Karpov – Kasparov
World Ch (game 9), Moscow 1984/5

Here Shredder 7 gives the PV 23 ♕a5 ♖e8 24 ♖d4 ♗g4 25 ♖xc4 ♕xc4 26 ♗f1 ♖c8 27 ♖d2 ♗e6 28 e4 ♕c5 29 ♖xd5 ♗xd5 30 ♕d8+ 0.42/14. The first move is very reasonable; in fact it is the same move that Karpov played in the game. But check out the final position after 30 ♕d8+?? *(D)*.

B

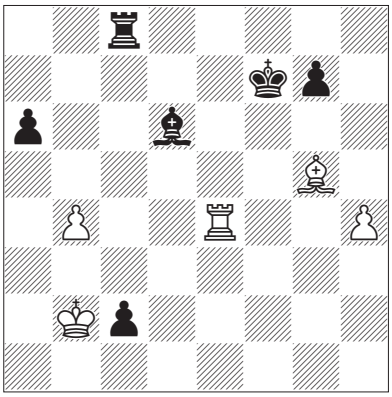


B



White has thrown away not only a rook, but decides the queen must be tossed away as well, all for no purpose whatsoever. Of course Shredder is an extremely strong program and would never actually play this line to the end. I am sure Shredder was not even evaluating the above position when it concluded that White is 0.42 pawns ahead. It is purely an artifact, and a rather bizarre one at that, of the way Shredder happens to generate its PVs, by trying to extract them after the fact from hash tables. The trouble is that hash-table contents can change between the time a program makes its evaluations and the time it tries to generate the PV.

B



One day, when analysing the above position with Shredder, it came up with an especially nonsensical PV: "Shredder 7.04: **44...g6 45 ♗d2 ♖c6 46 ♖g4 c1♕+ 47 ♗xc1 ♗e5+ 48 ♔b1 a5 49 bxa5 g5 50 hxg5 ♖c8 51 ♗b2 ♖c1+ 52 ♔xc1 ♗d4 53 ♖xd4** −1.81/24" *(D)*.

Black has given away, for no reason, all of his pieces, yet the evaluation shows Black is winning! I have even seen Shredder PVs where the PV ends in checkmate, yet Shredder's evaluation shows only a very slight advantage! These examples are neither unique, nor unusual. Particularly for programs such as Shredder (Shredder has notoriously bizarre PVs), which try to extract their main line of program analysis after the fact from hash tables, this is quite common. The moral of all this is to **never** believe a computer PV all the way to the end. The first move of a PV will always be what the program thinks is the best move, but after that, all bets are off. There are some programs, Crafty for example, which are careful only to display lines in the PV that were the same lines the program actually evaluated, but unless you are **sure** you are using such a program, **never** trust the full PV, and even then you need to double check it.

Another extremely useful interactive technique for chess analysis is running an engine while stepping **backwards** through the analysed variations. Of course, this only works when you already have analysis that you want to test, but in such cases it can be a **huge** timesaver, especially in positions where one side has limited choices: either tactically sharp and forcing lines, or positions where one side is in a terrible bind, without counterplay. It can also help in situations involving king attacks. The reason this works so well is that if a program sees a winning position, many programs can retain this information in their hash tables when backing up through the analysis. The method is