Formal Verification of Chess Endgame Databases

A case study in combining theorem proving and model checking

Joe Hurd

Computing Laboratory
Oxford University

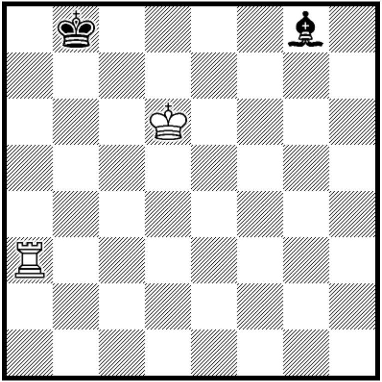
Emerging Trends
TPHOLs 2005

Chess Endgame Databases

- Can solve certain classes of chess endgame by enumerating all positions in a database.
  - Compute depth to mate by working backwards from the checkmate positions.
  - Ken Thompson solved most five piece endgames, and the state of the art is now six piece endgames.
- Combine theorem proving and model checking to construct a verified endgame database:
  - model checking provides an automatic algorithm to construct the set of winning positions;
  - and implementing this algorithm in a theorem prover results in a theorem that the endgame database logically follows from the rules of chess.

Verified Endgame Databases: Algorithm

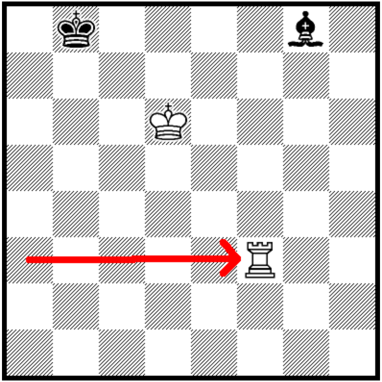
- Build a verified endgame database by working backwards from checkmates, but symbolically using BDDs.
- When computing the set of positions won in $n+1$ moves in a category C must consider the set of positions won in n moves in all the categories that can be reached from C in one move.
- Work up from the smaller categories to the bigger ones, iterating to a fixed point to compute the winning sets.
- Subtlety: Even though a fixed point is reached in 7 moves for King and two Rooks versus King, must still iterate 16 moves back because that was necessary for King and Rook versus King to converge!



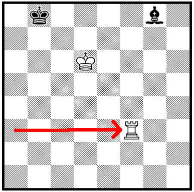
One White move is checkmate in 29, all other moves draw.

What is the winning move?

Verified Endgame Databases

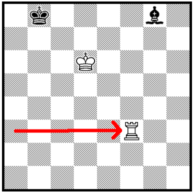


Rf3!



The result of querying our verified endgame database on this position:

$$
\begin{array}{ll}
\vdash & (\mathsf{Black}, \\
& \ \lambda sq. \\
& \quad \text{if } sq = (3,5) \text{ then SOME (White, King)} \\
& \quad \text{else if } sq = (5,2) \text{ then SOME (White, Rook)} \\
& \quad \text{else if } sq = (1,7) \text{ then SOME (Black, King)} \\
& \quad \text{else if } sq = (6,7) \text{ then SOME (Black, Bishop)} \\
& \quad \text{else NONE)} \ \in \ \text{win2\_by chess } 28 \ \wedge \ \cdots
\end{array}
$$



In fact, checkmate in 29 is the longest possible win in the King and Rook versus King and Bishop endgame.

$$
\begin{array}{l}
\vdash \quad \forall p.\\
\qquad \text{all\_on\_board}\ p \ \wedge\ \text{to\_move}\ p = \text{White}\ \wedge\\
\qquad \text{has\_pieces}\ p\ \text{White}\ [\text{King}; \text{Rook}]\ \wedge\\
\qquad \text{has\_pieces}\ p\ \text{Black}\ [\text{King}; \text{Bishop}]\ \Longrightarrow\\
\qquad p \in \text{win1 chess} \iff p \in \text{win1\_by chess } 28
\end{array}
$$

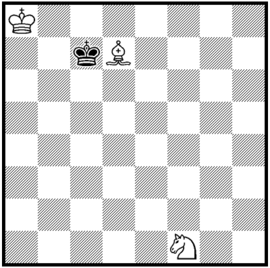
Application 1: Golden Reference Endgame Database

- The state of the art in endgame database correctness is summed up in the following quotation:

  "Both [Nalimov's endgame databases] and those of Wirth yield exactly the same number of mutual zugzwangs [...] for all 2-to-5 man endgames and no errors have yet been discovered."
- Improvement: our verified endgame database logically follows from the rules of chess.
- Can use as a golden reference to test other endgame databases:
  - randomly sample positions to check evaluation;
  - and also compute global properties such as the number of positions of a certain type (BDD computation).

Application 2: Teaching Aid for Chess Beginners

- Have used the verified endgame database to create some educational web pages showing the best lines of defence.
- Example: Checkmating a bare King with King, Bishop and Knight is something that beginners struggle to learn.



$\rightsquigarrow$

33
moves
later

$\rightsquigarrow$