Algorithms for solving sequential (zero-sum) games

Main case in these slides: chess

Slide pack by
Tuomas Sandholm
Figure 5.12 Ratings of human and machine chess champions.

Deep Blue team. Front, left to right: Joel Benjamin, Chung-Jan Tan. Back, left to right: Jerry Brady, Murray Campbell, Feng-Hsiao Hsu, and Joe Hoane.

1997
3½ - 2½
Loss-win-draw-draw-draw-win
Rich history of cumulative ideas

Claude Shannon, Alan Turing
Kotok/McCarthy Program
& ITEP Program
MAC HACK
Chess 3.0—Chess 4.9
Belle
Cray Blitz
Hitech
Deep Blue

Minimax search with scoring function 1950
Alpha-beta search, brute force search 1966
Transposition tables 1967
Iteratively-deepening depth-first search 1975
Special-purpose circuitry 1978
Parallel search 1983
Parallel evaluation 1985
Parallel search and special-purpose circuitry 1987
Quiescence search 1960's?
End game databases via dynamic programming 1977
Conspiracy numbers 1988
Singular extension 1980's
Opening books
Evaluation function learning & engineering 1950's
Game-theoretic perspective

- Game of perfect information
- Finite game
  - Finite action sets
  - Finite length
- Chess has a solution: win/tie/lose (Nash equilibrium)
- Subgame perfect Nash equilibrium (via backward induction)
- REALITY: computational complexity bounds rationality
Chess game tree

- Initial position
- 20 positions after White's first move
- Opening stage: Databases for opening moves usually cover the first 5-15 moves
- 400 positions after one move by each side
- Databases for all 5 and some 6 piece endgames
- Endgame stage
- Middlegame stage: Moves in the middlegame are selected by carrying out a large search guided by the minimax algorithm

The search tree fans out at an average of 30-40 moves at each position in the tree.
Opening books (available on CD)

Example opening where the book goes 16 moves (32 plies) deep

Ruy Lopez

Marshall (Counter) Attack

1 e4 e5 2 Nf3 Nc6 3 Bb5 a6 4 Ba4 Nf6 5 0–0 Be7 6 Re1
b6 7 Bb3 0–0 8 c3 d5 9 exd5

<table>
<thead>
<tr>
<th></th>
<th>97</th>
<th>98</th>
<th>99</th>
<th>100</th>
<th>101</th>
<th>102</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Nxe5</td>
<td>Nxe5</td>
<td>dxc6(p)</td>
<td>exf3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Rxe5</td>
<td>c6!</td>
<td>d4l(q)</td>
<td>fxg2(r)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>d4……………..Bxd5 ………g3(b)</td>
<td>d4</td>
<td>Qh3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Bd6</td>
<td>cxd5</td>
<td>Bd6(l)</td>
<td>Bd6</td>
<td>Be6</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Re1…Re2</td>
<td>d4</td>
<td>Re1</td>
<td>Re1</td>
<td>Bf4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Qh4</td>
<td>Bg4(c)</td>
<td>B6d</td>
<td>Qd7(l)</td>
<td>Ng4</td>
<td>Nd5</td>
</tr>
<tr>
<td>14</td>
<td>g3</td>
<td>f3</td>
<td>Re3</td>
<td>d3</td>
<td>h3</td>
<td>Kg3</td>
</tr>
<tr>
<td></td>
<td>Qh3</td>
<td>Bh5</td>
<td>Qh4(f)</td>
<td>Qh3</td>
<td>Qh4(m)</td>
<td>a5</td>
</tr>
<tr>
<td>15</td>
<td>Be3(a)</td>
<td>Bxd5(d)</td>
<td>h3</td>
<td>Re4</td>
<td>Qf3</td>
<td>Nd2 +</td>
</tr>
<tr>
<td></td>
<td>Bg4</td>
<td>cxd5</td>
<td>Q4</td>
<td>Qf5</td>
<td>Nxf2</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Qd3</td>
<td>Nd2</td>
<td>Re5</td>
<td>Nd2</td>
<td>Re2(n)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Rxe8(b)</td>
<td>Qe7(e)</td>
<td>Qg6(g)</td>
<td>Qg6(k)</td>
<td>Ng4(o)</td>
<td></td>
</tr>
</tbody>
</table>

(a) 15 Re4? g5 16 Qf3 (16 Bg5?? Qg5) 18 … Bfs 17 Be2 (17 Bh4??) 17 … Bxe4 18 Bxe4 Qe6
19 Bg5 (19 Bf5! Qe7 20 Kg2 Qxc1 21 Na3 Qf2 wins) 19 … f5 20 Bd3 h6 1 (Gutman).
(b) Short–Pinter, Rotterdam, 1986 continued 17 Nd2 Re8 18 a4 bxa4 19 Rxa4 f5 20 Qf1 Qh5
21 a5 Re8 22 Bxb5 cxb5 23 Re6 Rxe6 24 Qe6 Qh7 25 h3! with complications favoring
White.
(c) 17 … Qh4 14 g3 Qh5 (14 … Qh6 15 Nd2 Bf5 16 Ne4??) 15 Nd2 Bg4 16 f3 Bxf3 17 Nxf3
Qxf3 18 Rf2 Qe4 19 Qf3 2, Sax–P. Nikolic, Plovdiv 1983.
(d) If 15 Nd2 Ne4 is annoying.
(e) 17 Nf1 Re8 18 Be3 Qe4 2, van der Sterren–Pevn, Brussels 1984. Black has good play for
the pawn.
(f) 14 … f5 15 Nd2 f4 16 Re1 Qf5 17 Nf3 Qh5 18 Ne5 f3 19 gxf3 Bxh3 20 f4 ± (Tal).
(g) 17 Re1 Qg6 18 Qf3 Ne8 19 Bf4 Bxd4 20 Qxd4 Bxh3 21 Qg3 Qxg3 =, Tal–Spassky, match
1965.
(h) 12 d3 B6d 13 Re1 [13 … Qb4 14 g3 Qh3 transposes back into the column] 13 … Bfs!
14 Nd2 Nfs 15 Ne4 Nxd3 16 Bg5 Qf7 17 Re3 Re8 18 Bxe4 Rxe8 =, Kf. Georgiev–Nunn,
Dubai 1996.
(i) Cellier’s 12 … Bfs 13 Re1 e5 14 d4 Bb7, playing for central control, is a reasonable
alternative.
(j) 13 … Nf6 14 d4 Bg4 15 Qd3 c5 16 Bc2 is better for White, according to Fischer.
Minimax algorithm (not all branches are shown)

1. Draw tree to a depth of two levels
   (Note 1: not all moves at the first two levels are shown in this figure)

2. Assign a score to each terminal position.
   (Note 2: only the material on the board is used to assign
   scores to the four positions shown. In the top one, White has
   a three pawn advantage so a score of +3.)

3. Assign backed-up scores to the non-terminal positions.

4. Determine the principal continuation
   (Shown bold in this figure)

5. Select the move to play as the first move on this
   continuation.
Deeper example of minimax search

ABJKL is equally good
recursive function MINIMAX(POSITION, DEPTH);
{MINIMAX is the name of the process, which requires two inputs: a chess POSITION with white to move, and a number DEPTH indicating the ply level at which evaluation is to take place. The result of this process is the minimax value of the position}
if
    DEPTH = 0
then
    MINIMAX := EVAL(POSITION)
    {the function EVAL evaluates at the bottom level}
else
    begin
        MINIMAX := FINDMOVES(POSITION, MOVES, NMOVES)
        {the move generator finds all legal moves from POSITION; the value produced and stored in MINIMAX is that of a loss, say -100, or zero if stalemate (NMOVES = 0 and no check)}
        if
            NMOVES > 0
        then
            for
                i := 1 to NMOVES do
                NEWPOSITION := SWAPSIDES(MAKEMOVE(POSITION, MOVE(i)));
                {produces a new position, by making move i in POSITION, and then reversing Black and White sides}
                VALUE := -MINIMAX(NEWPOSITION, DEPTH-1);
                {here comes the magic: assuming that the MINIMAX function is available for use (not quite true at the time this line is written), it is called upon to produce a minimax value for NEWPOSITION (with depth decreased by 1); since this value is with respect to the Black side, its sign is reversed}
        if
            VALUE > MINIMAX
        then
            MINIMAX := VALUE
        {MINIMAX contains the largest value found up to now; in this example, no record is kept of the associated move}
end do
end

Folk wisdom for playing against computers:
Play open positions ⇒ increases the branching factor
⇒ reduces computer's lookahead.
Search depth pathology

- Beal (1980) and Nau (1982, 83) analyzed whether values backed up by minimax search are more trustworthy than the heuristic values themselves. The analyses of the model showed that backed-up values are somewhat less trustworthy.
- Anomaly goes away if sibling nodes’ values are highly correlated [Beal 1982, Bratko & Gams 1982, Nau 1982].
- Pearl (1984) partly disagreed with this conclusion, and claimed that while strong dependencies between sibling nodes can eliminate the pathology, practical games like chess don’t possess dependencies of sufficient strength.
  - He pointed out that few chess positions are so strong that they cannot be spoiled abruptly if one really tries hard to do so.
  - He concluded that success of minimax is “based on the fact that common games do not possess a uniform structure but are riddled with early terminal positions, colloquially named blunders, pitfalls or traps. Close ancestors of such traps carry more reliable evaluations than the rest of the nodes, and when more of these ancestors are exposed by the search, the decisions become more valid.”
- Still not fully understood. For new results, see:
$\alpha - \beta$ - pruning

Partially drawn game tree showing deep alpha-beta cutoff
α-β -search on ongoing example
\[ \alpha - \beta \text{-search} \]

function \( \text{MAX-VALUE}(\text{state}, \text{game}, \alpha, \beta) \) returns the minimax value of \( \text{state} \)

inputs: \( \text{state} \), current state in game

\( \text{game} \), game description

\( \alpha \), the best score for \( \text{MAX} \) along the path to \( \text{state} \)

\( \beta \), the best score for \( \text{MIN} \) along the path to \( \text{state} \)

if \( \text{CUTOFF-TEST}(\text{state}) \) then return \( \text{EVAL}(\text{state}) \)

for each \( s \) in \( \text{SUCCESSORS}(\text{state}) \) do

\[ \alpha \leftarrow \text{MAX}(\alpha, \text{MIN-VALUE}(s, \text{game}, \alpha, \beta)) \]

if \( \alpha \geq \beta \) then return \( \beta \)

end

return \( \alpha \)

function \( \text{MIN-VALUE}(\text{state}, \text{game}, \alpha, \beta) \) returns the minimax value of \( \text{state} \)

if \( \text{CUTOFF-TEST}(\text{state}) \) then return \( \text{EVAL}(\text{state}) \)

for each \( s \) in \( \text{SUCCESSORS}(\text{state}) \) do

\[ \beta \leftarrow \text{MIN}(\beta, \text{MAX-VALUE}(s, \text{game}, \alpha, \beta)) \]

if \( \beta \leq \alpha \) then return \( \alpha \)

end

return \( \beta \)
# Complexity of α-β-search

<table>
<thead>
<tr>
<th>Search Depth (DMAX)</th>
<th>Best case Minimum number of terminal positions in an alpha-beta search</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>$2 \times 3^{10} \approx 6 \times 10^{1} = 60$</td>
</tr>
<tr>
<td>4</td>
<td>$2 \times 3^{20} \approx 2 \times 10^{3} = 2,000$</td>
</tr>
<tr>
<td>6</td>
<td>$2 \times 3^{30} \approx 6 \times 10^{4} = 60,000$</td>
</tr>
<tr>
<td>8</td>
<td>$2 \times 3^{40} \approx 2 \times 10^{6} = 2,000,000$</td>
</tr>
<tr>
<td>10</td>
<td>$2 \times 3^{50} \approx 6 \times 10^{7} = 60,000,000$</td>
</tr>
<tr>
<td>12</td>
<td>$2 \times 3^{60} \approx 2 \times 10^{9} = 2,000,000,000$</td>
</tr>
<tr>
<td>14</td>
<td>$2 \times 3^{70} \approx 6 \times 10^{10} = 60,000,000,000$ (Deep Blue)</td>
</tr>
<tr>
<td>16</td>
<td>$2 \times 3^{80} \approx 2 \times 10^{12}$ = 2,000,000,000,000</td>
</tr>
</tbody>
</table>

Best case: α-β allows search 9× as deep as minimax.
Worst case: α-β does not prune a single node.
Average case based on random order of moves $O(b^d) \rightarrow O((b \log b)^d)$
Close to best case by exploring better moves first
- captures $\rightarrow$ threats $\rightarrow$ forward moves $\rightarrow$ backward moves

Hash table $\rightarrow$ iterative deepening search and use backed up values from one iteration to determine the ordering of successors in the next iteration.
Variance in search time (due to α-β and quiescence search) $\Rightarrow$ iterative deepening (used by all major chess programs).
Evaluation function

- Difference (between player and opponent) of
  - Material
  - Mobility
  - King position
  - Bishop pair
  - Rook pair
  - Open rook files
  - Control of center (piecewise)
  - Others

Values of knight’s position in Deep Blue
Evaluation function...

- Deep Blue used ~6,000 different features in its evaluation function (in hardware)
- A different weighting of these features is downloaded to the chips after every real world move (based on current situation on the board)
  - Contributed to strong positional play
- Acquiring the weights for Deep Blue
  - Weight learning based on a database of 900 grand master games (~120 features)
    - Alter weight of one feature => 5-6 ply search => if matches better with grand master play, then alter that parameter in the same direction further
    - Least-squares with no search
  - Other learning is possible, e.g. Tesauro’s Backgammon
    - Solves credit assignment problem
    - Was confined to linear combination of features
  - Manually: Grand master Joel Benjamin played take-back chess. At possible errors, the evaluation was broken down, visualized, and weighting possibly changed
Databases of expert games

Deep Blue does not use these during play
Deep Blue uses them offline to learn evaluation.

KUPREJČIK 2520
VLADO KOVAČEVIĆ 2545
Ljubljana/Rogaška Slatina 1989


(diagram)

24... ḅc8? [24... ḅc1! 25. ḅg7 ḅf8 a)] 26. ḅg4 ḅa6 (26... ḅg5 27. e6 ḅa7 28. ḅe5 ḅd8 29. f7 ḅd8 30. ḅc6±) 27. ḅf6

KUPREJČIK 2520 — KOSTEN 2505
Torcy 1989

1. e4 e6 2. d4 d5 3. e5 c5 4. c3 dxe7 5. ḅf3 ḅd7 6. ḅe2 [RR 6. ḅd3 ḅg7 7. 0–0 cd4 8. cd4 ḅc8 N 9. ḅc3 ḅe7 10. ]
Horizon problem

A series of checks by the black rook forces the inevitable queening move by white “over the horizon” and makes this position look like a slight advantage for black, when it is really a sure win for white.
Ways to tame the horizon effect

• Quiescence search
  – Evaluation function (domain specific) returns another number in addition to evaluation: stability
    • Threats
    • Other
  – Continue search (beyond normal horizon) if position is unstable
  – Introduces variance in search time

• Singular extension
  – Domain independent
  – A node is searched deeper if its value is much better than its siblings’
  – Even 30-40 ply
  – A variant is used by Deep Blue
Transpositions
Transpositions are important

<table>
<thead>
<tr>
<th>Depth of Search</th>
<th>Terminal positions in tree</th>
<th>Number of different terminal positions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>15</td>
<td>3x5 = 15</td>
</tr>
<tr>
<td>3</td>
<td>90</td>
<td>9x5 = 45</td>
</tr>
<tr>
<td>4</td>
<td>405</td>
<td>9x8 = 72</td>
</tr>
<tr>
<td>5</td>
<td>~2,000</td>
<td>13x8 = 112</td>
</tr>
<tr>
<td>6</td>
<td>~10,000</td>
<td>13x10 = 140</td>
</tr>
<tr>
<td>7</td>
<td>~50,000</td>
<td>17x10 = 170</td>
</tr>
<tr>
<td>8</td>
<td>~250,000</td>
<td>17x12 = 204</td>
</tr>
<tr>
<td>9</td>
<td>~1,250,000</td>
<td>~25x16 ~ 400</td>
</tr>
<tr>
<td>10</td>
<td>~6,250,000</td>
<td>~25x25 ~ 625</td>
</tr>
</tbody>
</table>
Transposition table

• Store millions of positions in a hash table to avoid searching them again
  – Position
  – Hash code
  – Score
  – Exact / upper bound / lower bound
  – Depth of searched tree rooted at the position
  – Best move to make at the position

• Algorithm
  – When a position P is arrived at, the hash table is probed
  – If there is a match, and
    • new_depth(P) ≥ stored_depth(P), and
    • score in the table is exact, or the bound on the score is sufficient to cause the move leading to P to be inferior to some other choice
  – then P is assigned the attributes from the table
  – else computer scores (by direct evaluation or search (old best move searched first)) P and stores the new attributes in the table

• Fills up => replacement strategies
  – Keep positions with greater searched tree depth under them
  – Keep positions with more searched nodes under them
Search tree illustrating the use of a transposition table

1. Move x refutes move z and position T is placed in the transposition table with an upper bound of 36 assigned to its score. Move y is never examined. When search of the subtree beginning with move s is finished, the root is assigned a score of 26.

2. When position T is arrived at the second time, it is found in the transposition table. The upper bound of 36 is enough to consider T a terminal position.

3. The third time position T arises, the bound of 36 is insufficient to call T a terminal position. Its successors are searched and a score of 32 is stored in the transposition table along with move y, which was found best. Move v is the best searched thus far.

4. The fourth time position T arises, it is at a shallower level in the tree than previously. Its score cannot be used, but move y, found best the last time position T was searched, is ordered to be searched first this time. A backed-up score of 35 is assigned to T when search of the subtree rooted at T is finished. When search terminates, the root will have a score of 35.
End game databases

Torres y Quevedo's Mating Algorithm

Torres' scheme for effecting mate in the KRK endgame assumes an initial position with the automaton's White King on a8, Rook on b8, and the opponent's King on any unchecked square in the first six ranks. His algorithm for moving can be described in programming notation:

```
if both BK and R are on left side {files a,b,c}  
then move R to file h {keep R out of reach of K}  
elseif both BK and R are on right side {files f,g,h}  
then move rook to file a {keep R away from K}  
elseif rank of R exceeds rank of BK by more than one  
then move R down one rank {limit scope of BK}  
elseif rank of WK exceeds rank of BK by more than two  
then move WK down one {WK approaches to support R}  
elseif horizontal distance between kings is odd  
then {make tempo move with R}  
   if R is on a file then move R to b file  
   elseif R is on b file then move R to a file  
   elseif R is on g file then move R to h file  
   else {R is on h file} move R to g file  
endif  
elseif horizontal distance between kings is not zero  
then move WK horizontally toward BK {keep opposition}  
else give check by moving rook down  
   {and if on first rank, it's mate}  
endif
```

If the opponent's King is placed on a6, with best delaying tactics mate can be staved off for 61 moves.
Generating databases for solvable subgames

- State space = \{WTM, BTM\} x \{all possible configurations of remaining pieces\}
- BTM table, WTM table, legal moves connect states between these
- Start at terminal positions: mate, stalemate, immediate capture without compensation (=reduction). Mark white’s wins by won-in-0
- Mark unclassified WTM positions that allow a move to a won-in-0 by won-in-1 (store the associated move)
- Mark unclassified BTM positions as won-in-2 if forced moved to won-in-1 position
- Repeat this until no more labellings occurred
- Do the same for black
- Remaining positions are draws
Compact representation methods to help endgame database representation & generation

Squares for Black's king that must be considered in KRK database.

Building a KQK database: (a) initial contents of database, and (b) contents after performing the first step.
Endgame databases...

1977
Game 1
[Ken Thompson]
White: Walter Browne  Black: BELLE

Figure 6.17. Position from BELLE's database:
White to play and win in thirty moves.

Computer could hold a lost position against
IM Hans Berliner.

Separated rook & king.

Folks wisdom of playing open positions?
Endgame databases...

KNNKP(d4) endgame with White to play and win

1 Nb4+ Kb6 2 Nd3 Kc7 3 Nb5+ Kc6 4 Na3 Kb6 5 Kb8 (5 Nc4+ or 5 Nc2) Kc6
6 Nc4 (6 Nc2) Kb5 7 Nce5 Kb6 8 Kc8 Ka6 (8 ... Ka5 or 8 ... Kb5) 9 Kc7 (9 Kd7) Kb5
10 Kd6 Ka4 11 Kc5 Kb3 12 Kb5 Kc3 13 Ka4 Kc2 14 Kd4 Kd1 15 Kb3 Kd2 16 Kb2 Ke1
17 Nc4 Ke2 18 Kc2 Kf3 19 Kd2 (19 Kd1) Kg3 (19 ... Ke4) 20 Ke2 (20 Nc5) Kg2
21 Nce5 Kg3 22 Kf1 Kh4 23 Kg2 (23 Kf2) Kg5 24 Kf3 Kf5 25 Nc4 Kf6 26 Kf4 Ke6
27 Ke4 Kf6 28 Kd5 Ke7 29 Ke5 Kf7 30 Kd6 Kf6 31 Nd2 Kf5 32 Ke7 Kg6 33 Ke6 Kg7
(33 ... Kg5) 34 Ne4 Kg6 35 Ke5 Kd6 36 Kd6 Kh7 (36 ... Kh6) 37 Nd2 (37 Ne2) Kg7
38 Ke6 Kf8 39 Ne4 (39 Ng4) Ke8 40 Nf6+ (40 Nd6+) Kf8 (40 ... Kd8) 41 Nh5 Ke8
42 Ng7+ Kd8 43 Kd6 Kc8 44 Ne6 Kb8 (44 ... Kf7) 45 Kc5 Ka7 46 Kc6 Ka6 47 Nec5+
(47 Ng5) Ka5 48 Nb3+ (48 Ne4) Ka4 49 Nd2 Ka5 50 Kc5 Ka6 51 Nc4 Kb7 52 Kd6 Kc8
53 Na5 Kd8 54 Nb7+ Ke8 55 Ke6 Kf8 56 Nd6 Kg7 57 Kf5 Kh6 58 Kf6 Kh5 59 Nf7
(59 Ne4) Kg4 60 Ng5 Kh4 61 Kf5 Kg3 62 Ke4 Kg4 63 Nf7 Kh5 (63 ... Kg3) 64 Kf5 Kh4
65 Nf5 Kh5 66 Ng4 Kh4 67 Nf6 Kh3 68 Ke5 Kg3 69 Ke4 Kh3 70 Kf3 Kh4 71 Kf4 Kh3
72 Ne8 (72 Ne4 or 72 Nh4) Kh4 73 Ng7 Kh3 74 Nf5 Kg2 (74 ... Kh2) 75 Kg4 Kh2
(75 ... Kf1 or 75 ... Kg1 or 75 ... Kh1) 76 Nd6 (76 Ng3) Kg2 (76 ... Kg1 or 76 ... Kh1)
77 Nc4 (77 Ne4) Kh2 (77 ... Kg1) 78 Nd2 Kg2 79 Kh4 Kh2 (79 ... Kg1) 80 Nf4
(80 Ne1) Kg1 81 Kg3 Kh1 82 Nf3 (82 Ne2 or 82 Nh3) d3 followed by 83 Nh3 d2
84 Nf2#.
How end game databases changed chess

• All 5 piece endgames solved (can have $> 10^8$ states) & many 6 piece
  – KRBKNN ($\sim 10^{11}$ states): longest path-to-reduction 223
• Rule changes
  – Max number of moves from capture/pawn move to completion
• Chess knowledge
  – Splitting rook from king in KRKQ
  – KRKN game was thought to be a draw, but
    • White wins in 51% of WTM
    • White wins in 87% of BTM
Endgame databases...

<table>
<thead>
<tr>
<th>Three Pieces</th>
<th>Four Pieces</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Endgame</strong></td>
<td><strong>Maximum number of moves to win</strong></td>
</tr>
<tr>
<td>KQK</td>
<td>10 to mate</td>
</tr>
<tr>
<td>KRK</td>
<td>16 to mate</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Five Pieces</th>
<th>Maximum number of moves to a win (mate or conversion)</th>
<th>Maximum number of moves to a win (mate or conversion)</th>
<th>Maximum number of moves to a win (mate or conversion)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Endgame</strong></td>
<td><strong>Endgame</strong></td>
<td><strong>Endgame</strong></td>
<td><strong>Endgame</strong></td>
</tr>
<tr>
<td>KNNNK</td>
<td>21</td>
<td>KBBKQ</td>
<td>4</td>
</tr>
<tr>
<td>KNNBK</td>
<td>14</td>
<td>KBRKN</td>
<td>21</td>
</tr>
<tr>
<td>KNRK</td>
<td>11</td>
<td>KBBKB</td>
<td>25</td>
</tr>
<tr>
<td>KNNQK</td>
<td>7</td>
<td>KBRKR</td>
<td>59</td>
</tr>
<tr>
<td>KNNKN</td>
<td>7</td>
<td>KBBKQ</td>
<td>7</td>
</tr>
<tr>
<td>KNNKB</td>
<td>4</td>
<td>KBBK</td>
<td>7</td>
</tr>
<tr>
<td>KNNKR</td>
<td>3</td>
<td>KBQK</td>
<td>8</td>
</tr>
<tr>
<td>KNNKQ</td>
<td>1</td>
<td>KBRKR</td>
<td>19</td>
</tr>
<tr>
<td>KNBKN</td>
<td>77</td>
<td>KBKQ</td>
<td>30</td>
</tr>
<tr>
<td>KNBKB</td>
<td>13</td>
<td>KRQK</td>
<td>8</td>
</tr>
<tr>
<td>KNBKR</td>
<td>6</td>
<td>KBBQ</td>
<td>5</td>
</tr>
<tr>
<td>KNBKQ</td>
<td>5</td>
<td>KBBQ</td>
<td>5</td>
</tr>
<tr>
<td>KRKNK</td>
<td>24</td>
<td>KRQK</td>
<td>5</td>
</tr>
<tr>
<td>KRKNR</td>
<td>25</td>
<td>KRBB</td>
<td>6</td>
</tr>
<tr>
<td>KRKQ</td>
<td>33</td>
<td>KRKN</td>
<td>6</td>
</tr>
<tr>
<td>KNNQK</td>
<td>9</td>
<td>KRRK</td>
<td>5</td>
</tr>
<tr>
<td>KNNKH</td>
<td>9</td>
<td>KRRK</td>
<td>4</td>
</tr>
<tr>
<td>KNNKB</td>
<td>9</td>
<td>KRKN</td>
<td>8</td>
</tr>
<tr>
<td>KNNKR</td>
<td>22</td>
<td>KRKB</td>
<td>10</td>
</tr>
<tr>
<td>KNNQK</td>
<td>35</td>
<td>KRKQ</td>
<td>25</td>
</tr>
<tr>
<td>KNNKB</td>
<td>14</td>
<td>KQKB</td>
<td>16</td>
</tr>
<tr>
<td>KBKBN</td>
<td>11</td>
<td>KRRK</td>
<td>5</td>
</tr>
<tr>
<td>KBBBK</td>
<td>11</td>
<td>KKKBN</td>
<td>5</td>
</tr>
<tr>
<td>KBBRK</td>
<td>6</td>
<td>KBBQK</td>
<td>16</td>
</tr>
<tr>
<td>KBKKB</td>
<td>6</td>
<td>KBKQ</td>
<td>60</td>
</tr>
<tr>
<td>KBKKB</td>
<td>6</td>
<td>KRKNN</td>
<td>11</td>
</tr>
<tr>
<td>KBKKB</td>
<td>7</td>
<td>KRKNB</td>
<td>13</td>
</tr>
</tbody>
</table>

Figure 6.14. On the maximum number of moves to force a win in endgames with no more than five pieces other than pawns.
Deep Blue’s search

- ~200 million moves / second = 3.6 * 10^10 moves in 3 minutes
- 3 min corresponds to
  - ~7 plies of uniform depth minimax search
  - 10-14 plies of uniform depth alpha-beta search
- 1 sec corresponds to 380 years of human thinking time
- Software searches first
  - Selective and singular extensions
- Specialized hardware searches last 5 ply
Deep Blue’s hardware

- 32-node RS6000 SP multicomputer
- Each node had
  - 1 IBM Power2 Super Chip (P2SC)
  - 16 chess chips
    - Move generation (often takes 40-50% of time)
    - Evaluation
    - Some endgame heuristics & small endgame databases
- 32 Gbyte opening & endgame database
Role of computing power

Figure 6.23. Relationship between the level of play by chess programs and the size of the tree searched during a three minute move.

Figure 6.24. Results of Thompson's two experiments: (a) first experiment, (b) second experiment. Entries in the tables indicate the number of games won by the program heading the row against the program heading the column.

Figure 6.25. Percentage of time Bell(1) picked different moves from Bell(1) and the corresponding predicted ratings based on expression (1) for two cases: (1) \( R(4) = 1320 \) and \( R(5) = 1570 \), and (2) \( R(4) = 1300 \) and \( R(5) = 1570 \).

Diminishing returns to computation power.
Kasparov lost to Deep Blue in 1997

- Win-loss-draw-draw-draw-loss
  - (In even-numbered games, Deep Blue played white)
Future directions

• Engineering
  – Better evaluation functions for chess
  – Faster hardware
  – Empirically better search algorithms
  – Learning from examples and especially from self-play
  – There already are grandmaster-level programs that run on a regular PC, e.g., Fritz

• Fun
  – Harder games, e.g. Go
  – Easier games, e.g., checkers (some openings solved [2005])

• Science
  – Extending game theory with normative models of bounded rationality
  – Developing normative (e.g. decision theoretic) search algorithms
    • MGSS* [Russell&Wefald 1991] is an example of a first step
    • Conspiracy numbers

• Impacts are beyond just chess
  – Impacts of faster hardware
  – Impacts of game theory with bounded rationality, e.g. auctions, voting, electronic commerce, coalition formation