Search Space Size Reductions

**Worst Case:** In an ordering where worst options evaluated first, all nodes must be examined.

**Best Case:** If nodes ordered so that the best options are evaluated first, then what?

---

The Need for Imperfect Decisions

**Problem:** Minimax assumes the program has time to search to the terminal nodes.

**Solution:** Cut off search earlier and apply a heuristic evaluation function to the leaves.

---

Static Evaluation Functions

Minimax depends on the translation of board quality into a single, summarizing number. Difficult. Expensive.

- Add up values of pieces each player has (weighted by importance of piece).
- Isolated pawns are bad.
- How well protected is your king?
- How much maneuverability do you have?
- Do you control the center of the board?
- Strategies change as the game proceeds.

---

Design Issues for Heuristic Minimax

**Evaluation Function:** Need to be carefully crafted and depends on game! What criteria should an evaluation function fulfill?

---

Linear Evaluation Functions

- $w_1 f_1 + w_2 f_2 + ... + w_n f_n$
- This is what most game playing programs use

Steps in designing an evaluation function:
1. Pick informative features
2. Find the weights that make the program play well

---

Design Issues for Heuristic Minimax

**Search:** search to a constant depth

What are problems with constant search depth?
Backgammon – Rules

- Goal: move all of your pieces off the board before your opponent does.
- Black moves counterclockwise toward 0.
- White moves clockwise toward 25.
- A piece can move to any position except one where there are two or more of the opponent’s pieces.
- If it moves to a position with one opponent piece, that piece is captured and has to start its journey from the beginning.

- If you roll doubles you take 4 moves (example: roll 5,5, make moves 5,5,5,5).
- Moves can be made by one or two pieces (in the case of doubles by 1, 2, 3 or 4 pieces)
- And a few other rules that concern bearing off and forced moves.

White has rolled 6-5 and has 4 legal moves: (5-10,5-11), (5-11,19-24), (5-10,10-16) and (5-11,11-16).

Expectiminimax

\[
\text{Expectiminimax} \ (n) = \begin{cases} 
\text{utility}(n) & \text{for } n, \text{ a terminal state} \\
\max_{s \in \text{Succ}(n)} \text{expectiminimax}(s) & \text{for } n, \text{ a Max node} \\
\min_{s \in \text{Succ}(n)} \text{expectiminimax}(s) & \text{for } n, \text{ a Min node} \\
\sum_{s \in \text{Succ}(n)} P(s) \ast \text{expectiminimax}(s) & \text{for } n, \text{ a chance node}
\end{cases}
\]
**State of the Art in Backgammon**

- 1980: *BKG* using two-ply (depth 2) search and lots of luck defeated the human world champion.
- 1992: Tesauro combines Samuel's learning method with neural networks to develop a new evaluation function (search depth 2-3), resulting in a program ranked among the top 3 players in the world.

**State of the Art in Checkers**

- 1952: Samuel developed a checkers program that learned its own evaluation function through self-play.
- 1990: *Chinook* (J. Schaeffer) wins the U.S. Open. At the world championship, Marion Tinsley beat *Chinook*.
- 2005: Schaeffer et al. solved checkers for “White Doctor” opening (draw) (about 50 other openings).

**State of the Art in Go**

Large branching factor makes regular search methods inappropriate.

Best computer Go programs ranked only “weak amateur”.

Employ pattern recognition techniques and limited search.

$2,000,000$ prize available for first computer program to defeat a top level player.

**History of Chess in AI**

<table>
<thead>
<tr>
<th>Year</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>1950s</td>
<td>Shannon and Turing both had programs that (barely) played legal chess (500 rank).</td>
</tr>
<tr>
<td>1950s</td>
<td>Alex Bernstein’s system, (500+ε).</td>
</tr>
<tr>
<td>1957</td>
<td>Herb Simon claims that a computer chess program would be world chess champion in 10 years...yeah, right.</td>
</tr>
<tr>
<td>1968</td>
<td>McCarthy, Michie, Papert bet Levy (rated 2325) that a computer program would beat him within 10 years.</td>
</tr>
<tr>
<td>1973</td>
<td>By 1973...Slate:“It had become too painful even to look at Chess 3.6 any more, let alone work on it.”</td>
</tr>
<tr>
<td>1973</td>
<td>Chess 4.0: smart plausible-move generator rather than...</td>
</tr>
</tbody>
</table>
speeding up the search. Improved rapidly when put on faster machines.

1976 Chess 4.5: ranking of 2070.


1980’s Programs depend on search speed rather than knowledge (2300 range).

1993 DEEP THOUGHT: Sophisticated special-purpose computer; $\alpha - \beta$ search; searches 10-ply; singular extensions; rated about 2600.

1995 DEEP BLUE: searches 14-ply; iterative deepening $\alpha - \beta$ search; considers 100–200 billion positions per move; regularly reaches depth 14; evaluation function has 8000+ features; singular extensions to 40-ply; opening book of 4000 positions; end-game database for 5-6 pieces.

1997 DEEP BLUE: first match won against world-champion (Kasparov).

2002 IBM declines re-match. FRITZ played world champion Vladimir Kramnik. 8 games. Ended in a draw.