Adversarial Search

Game Playing Between Agents
Games

- Two-agent, perfect information, two-player games
  - Two players move in turn until one wins (the other thereby loses), or result is draw
  - Perfect knowledge of environment
  - Idealized model of multiple-agent interaction, and maps exactly onto many common games, including chess
Search tree for a very boring robot game
Mini-max search terminology

- Two players: MAX and MIN
- Task is to find next “best” move for MAX
- MAX moves first (next), two players move alternately thereafter
- A ply of ply-depth $k$ in a game tree consists of nodes of depths $2k$ and $2k+1$
  - Extent of search in game tree usually given in terms of ply depth
Why chess is hard

- Estimate of number of nodes in complete search graph for chess:

  \[10^{40}\]

- Branching factor \(\approx 35\)
- Search to termination in such games is impossible (\(\approx 10^{22}\) centuries)
- This is why chess is an *empirical phenomenon* that has retained its fascination over centuries
Evaluation functions

- Best we can do: search to a specific depth, and apply static evaluation function to the state
  - Measures “worth” of being in that state
  - Adopt convention: if favorable to MAX, evaluation function is positive; if favorable to MIN, negative

- Evaluation function is game-specific, extracting features from the current state of the game
  - E.g., in chess, piece count is important, weighted by type of piece and position
The minimax procedure

- When it is MAX’s turn, assume MAX would choose best (highest) valued nodes among his options
- So backed-up value of a MAX node parent of MIN tip nodes is the maximum of static evaluations of tip nodes
- Backed-up value of MIN node parent of MAX tip nodes is minimum of children
Minimax procedure, cont.

- Mini-max applies static evaluation functions to the leaf nodes, and “backs-up” the values level by level through the game tree
  - What if the rival is not smart?
Tic-tac-toe example

Consider following evaluation function $e(p)$

- If $p$ is not a winning position for either player, $e(p) = (\text{number of complete rows, columns or diagonals still open for MAX}) - \text{number of complete rows, columns, diagonals open for MIN}$
- If $p$ is a win for MAX, $e(p) = +\infty$, if $p$ is a win for MIN, $e(p) = -\infty$
Tic-tac-toe example

MAX's move

Start node
Alpha-beta procedure

Basic ideas behind alpha-beta:

- Don’t separate generation of search tree and evaluation of nodes (do these together)
- If you know a move is going to be worse than the best alternative in hand, don’t waste time finding out how much worse it is
Example of a cutoff

Beta value = -1

Alpha value = -1

Start node

© 1998 Morgan Kaufman Publishers
Alpha-beta cutoff

- Maintain two threshold values
  - $\alpha$: Greatest lower bound on this side
  - $\beta$: Least upper bound on the rival side
  - *Cut off any branch* $< \alpha$ or $> \beta$
Efficiency of alpha-beta search

- Best-case for alpha-beta happens when nodes are ordered perfectly
  - Impossible to guarantee this!
- In such cases, number of leaf nodes generated is \( d^{\frac{d}{2}} \)
  - Or, in same time search *without* alpha-beta would search to depth \( d/2 \)
  - Or, reduces effective branching factor to about \( \sqrt{b} \)
- Happily, in practice, alpha-beta comes close to achieving optimal reduction
Other heuristics

- **Progressive Deepening**
  Analyze the situation at progressive depths
  \[
  \frac{\text{# of nodes at the leaf level}}{\text{# of nodes before the leaf level}} = b - 1
  \]
  $b$: branching factor

- **Singular extension**
  - Continue search if one move is much better (worse) than the rest
  - To avoid the *Horizon effect* caused by a fixed depth
Deep Blue vs. Gary Kasparov

- In May, 1997, Deep Blue beat Gary Kasparov 3.5 to 2.5 in a 6-game match
Features of *Deep Blue*

- Specific features of Deep Blue:
  - Alpha-beta minimax search
  - Progressive deepening for handling time constraints
  - Singular extension to handle the horizon effect
  - A complex static evaluation function, encoding ~6,000 features
  - Weighted evaluation function “tuned” automatically against a library of 900 grandmaster games, and manually against grandmaster Joel Benjamin
Deep Blue’s hardware

- 32 processor parallel computer
- 64G hard disk, with 32G devoted to opening and endgame database
- 512 customs IC’s
- Implemented in hardware:
  - Parallel alpha-beta search
  - Move generation (40-50% of effort)
  - Static evaluation (50 lookup tables in hardware)