Adversarial Search 2

CS51A
David Kauchak
Spring 2019

Some material borrowed from:
Sara Owsley Sood and others

Baby Nim
Take 1 or 2 at each turn
Goal: take the last match

What move should I take?

Admin
Assignment 10
Pre-registration

Baby Nim
Take 1 or 2 at each turn
Goal: take the last match

MAX wins
= 1.0

MIN wins/
MAX loses
= -1.0
Baby Nim

Take 1 or 2 at each turn
Goal: take the last match

MAX wins
\[ = 1.0 \]

MIN wins/
MAX loses

MAX wins
\[ = 1.0 \]

MIN wins/
MAX loses

MAX wins
\[ = 1.0 \]

MIN wins/
MAX loses

MAX wins
\[ = 1.0 \]

MIN wins/
MAX loses

MAX wins
\[ = 1.0 \]

MIN wins/
MAX loses

MAX wins
\[ = 1.0 \]

MIN wins/
MAX loses
Take 1 or 2 at each turn
Goal: take the last match

MAX wins
= 1.0

MIN wins/
MAX loses

MAX loses
= -1.0
Baby Nim

Take 1 or 2 at each turn
Goal: take the last match

MAX wins
\[= 1.0\]

MIN wins
\[= -1.0\]

MAX loses

MAX wins
\[= 1.0\]

MIN wins
\[= -1.0\]

MAX loses

Which move?
Baby Nim

Take 1 or 2 at each turn
Goal: take the last match

MAX wins
\[ = 1.0 \]

MIN wins/
MAX loses

Minimax example 2

Which move should be made: A1, A2 or A3?

Properties of minimax

Minimax is optimal!

Are we done?
Games State Space Sizes

On average, there are ~35 possible moves that a chess player can make from any board configuration...

<table>
<thead>
<tr>
<th>Branching Factor Estimates for different two-player games</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tic-tac-toe: 4</td>
</tr>
<tr>
<td>Connect Four: 7</td>
</tr>
<tr>
<td>Checkers: 10</td>
</tr>
<tr>
<td>Othello: 30</td>
</tr>
<tr>
<td>Chess: 35</td>
</tr>
<tr>
<td>Go: 300</td>
</tr>
</tbody>
</table>

CHINOOK (2007) can search entire space...

AlphaGo (created by Google), in April 2016 beat one of the best Go players:


What do we do?
**Alpha-Beta pruning**

An optimal pruning strategy
- only prunes paths that are suboptimal (i.e. wouldn’t be chosen by an optimal playing player)
- returns the same result as minimax, but faster

Games State Space Sizes

Pruning helps get a bit deeper
For many games, still can’t search the entire tree
Now what?

<table>
<thead>
<tr>
<th>Branching Factor Estimates for different two-player games</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tic-tac-toe</td>
</tr>
<tr>
<td>Connect Four</td>
</tr>
<tr>
<td>Checkers</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>computer-dominated</th>
</tr>
</thead>
<tbody>
<tr>
<td>Othello</td>
</tr>
<tr>
<td>Chess</td>
</tr>
<tr>
<td>Go</td>
</tr>
</tbody>
</table>

Go as deep as you can:
- estimate the score/quality of the state (called an evaluation function)
- use that instead of the real score
Tic Tac Toe evaluation functions

Ideas?

Example Tic Tac Toe EVAL

Tic Tac Toe
Assume MAX is using “X”

EVAL(state) =
if state is win for MAX:
   + ¥
if state is win for MIN:
   - ¥
else:
   (number of rows, columns and diagonals available to MAX) -
   (number of rows, columns and diagonals available to MIN)

= 6 - 4 = 2
= 4 - 3 = 1

Chess evaluation functions

Ideas?

Chess EVAL

Assume each piece has the following value:
- pawn = 1;
- knight = 3;
- bishop = 3;
- rook = 5;
- queen = 9;

EVAL(state) =
sum of the value of white pieces –
sum of the value of black pieces

= 31 - 36 = -5
Chess EVAL

Assume each piece has the following values:
- pawn = 1;
- knight = 3;
- bishop = 3;
- rook = 5;
- queen = 9;

\[ \text{EVAL(state)} = \text{sum of the value of white pieces} - \text{sum of the value of black pieces} \]

Any problems with this?

Chess EVAL

Ignores actual positions!

Actual heuristic functions are often a weighted combination of features:

\[ \text{EVAL}(s) = w_1 f_1(s) + w_2 f_2(s) + w_3 f_3(s) + \ldots \]

Chess EVAL

A feature can be any numerical information about the board:
- as general as the number of pawns
- to specific board configurations

Deep Blue: 8000 features!

history/end-game tables

History:
- keep track of the quality of moves from previous games
- use these instead of search

end-game tables:
- do a reverse search of certain game configurations, for example all board configurations with king, rook and king
- tells you what to do in any configuration meeting this criterion
- if you ever see one of these during search, you lookup exactly what to do
end-game tables

**Devastatingly good**

- Allows much deeper branching
  - for example, if the end-game table encodes a 20-move finish and we can search up to 14
  - can search up to depth 34

**Stiller (1996)** explored all end-games with 5 pieces
  - one case check-mate required 262 moves!

**Knoval (2006)** explored all end-games with 6 pieces
  - one case check-mate required 517 moves!

Traditional rules of chess require a capture or pawn move within 50 or it’s a stalemate

---

Opening moves

**At the very beginning, we’re the farthest possible from any goal state**

- People are good with opening moves
- Tons of books, etc. on opening moves
- Most chess programs use a database of opening moves rather than search

---

Nim

**K piles of coins**

- On your turn you must take one or more coins from one pile
- Player that takes the last coin wins

**Example:**
https://www.goobix.com/games/nim/