CS 491 CAP
Intro to Combinatorial Games

Jingbo Shang
University of Illinois at Urbana-Champaign
Nov 4, 2016
Outline

◊ What is combinatorial game?
◊ Example 1: Simple Game
◊ Zero-Sum Game and Minimax Algorithms
◊ Nim Game
◊ Recommended Readings
Outline

◊ What is combinatorial game?
◊ Example 1: Simple Game
◊ Zero-Sum Game and Minimax Algorithms
◊ Nim Game
◊ Recommended Readings
Combinatorial Games

◊ Turn-based
  ▪ There are two players moving alternately;
  ▪ Each turn, the player changes the current “state” using a valid “move”.

◊ Perfect Information
  ▪ There are no chance devices (e.g., dices) and both players have perfect information.

◊ The rules are such that the game must eventually end;
  ▪ At some state, there are no valid moves and the game ends at this point
  ▪ Can be a simple win-or-lose game, or involve points (no draw!)
  ▪ Note: no cycles or cycles are always not optimal!
Outline

◊ What is combinatorial game?
◊ **Example 1: Simple Game**
◊ Zero-Sum Game and Minimax Algorithms
◊ Nim Game
◊ Recommended Readings
Example 1: Game Setting

◊ Rules
  ▪ There are \( n \) stones in a pile.
  ▪ Two players take turns.
  ▪ Each turn, the player removes either 1 or 3 stones.
  ▪ The one who takes the last stones wins.

◊ Goal
  ▪ Find out the winner if both players play perfectly
  ▪ Perfectly means that
    • Players want to win!
    • Players are smart enough!
Example 1: State & Move

◊ State $x$
  ▪ the number of remaining stones in the pile

◊ Valid moves from state $x$
  ▪ If $x \geq 1$, $x \rightarrow (x - 1)$
  ▪ If $x \geq 3$, $x \rightarrow (x - 3)$

◊ State $x = 0$ is the losing state
  ◊ Because it has no valid move.
Example 1: Algorithm

◊ No cycles in the state transitions $\rightarrow$ dynamic programming
◊ $f(x)$ is a boolean value that whether the player starting with the state $x$ can win the game
◊ A player wins if there is a way to force the opponent to lose
  ▪ Conversely, a player loses if there is no such way
◊ $f(x) = \neg f(x - 1) \lor \neg f(x - 3)$
◊ State $x$ is the winning state if:
  ▪ $(x - 1)$ is the losing state OR $(x - 3)$ is the losing state
◊ Otherwise, $x$ is the losing state
◊ $O(n)$ solution get!
Example 1: More efficient?

◊ Let’s solve the first few cases with DP...
◊ DP tables for the first few values

<table>
<thead>
<tr>
<th></th>
<th>W/L</th>
<th>n</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>W/L</td>
<td>L</td>
<td>W</td>
<td>L</td>
<td>W</td>
<td>L</td>
<td>W</td>
<td>L</td>
<td>W</td>
</tr>
</tbody>
</table>

◊ What’s the pattern?
◊ Let’s prove our conjecture using induction
Example 1: Proof

◊ Conjecture:
  ▪ If $n$ is odd, the first player wins.
  ▪ Otherwise, (i.e., $n$ is even), the second player wins
◊ Clearly holds for $n = 0$
◊ $\forall n \geq 1$
  ▪ If $n$ is odd, the resulting number of stones after taking away 1 or 3 stones is always even
    • By the inductive argument, the next player loses, so the current player wins the game
  ▪ If $n$ is even, the resulting number of stones is always odd
    • By the inductive argument, the next player wins, so the current player loses the game
Outline

◊ What is combinatorial game?
◊ Example 1: Simple Game
◊ Zero-Sum Game and Minimax Algorithms
◊ Nim Game
◊ Recommended Readings
Zero-Sum Game: Game Setting

◊ Settings:
  ▪ Two players
  ▪ Zero-sum: If the first player’s score is $x$, the other player gets $-x$
  ▪ Each player tries to maximize his/her own score
  ▪ Both players play perfectly
◊ Can be solved using Minimax algorithm
Minimax Algorithm

- Recursive algorithm that decides the best move for the current player at a given state
- Let $f(S)$ be the optimal score of the current player who starts at state $S$
- Let $T_{S,1}, T_{S,2}, \ldots, T_{S,m_S}$ be states that can be reached from $S$ using a single move
- $f(S) = \max_{i=1}^{m_S} -f(T_{S,i})$
  - Intuition: minimizing the opponent’s score maximizes my score
Minmax Algorithm: Pseudocode

◊ Given state $S$, want to compute $f(S)$
◊ If we have computed $f(S)$
   ▪ Return $f(S)$ // Memoize (refer to DP lecture)
◊ Set $f(S) = -\infty$
◊ For $i = 1$ to $m_S$ do
   ▪ $f(S) = \max \left( f(S), -f(T_{S,i}) \right)$
◊ Return $f(S)$
Zero-Sum Game: Extension

◊ Points are associated with moves
◊ The game is not zero-sum
  ▪ Each player wants to maximize his own score
  ▪ Each player wants to maximize the difference between his score and the opponent’s
◊ There are more than two players

◊ All of the above can be solved using a similar idea
Example 2: Game Setting

◊ An array of \( n \) positive integers
◊ Two players take turns
◊ Each turn, the player can take a number at the either end of the array and add to his/her points and then the number disappears
◊ Players want to maximize their own scores
◊ If both play perfectly, output the score of each player
Example 2: State & Move

◊ State
  ▪ 
  (i, j) - the remaining numbers are from the i-th index to the j-th index
  ▪ f(i, j) is the optimal score for the current player at state (i, j)
  ▪ Let sum(i, j) be the sum of the numbers from the i-th index to the j-th index

◊ Move
  ▪ Take the i-th number: (i, j) → (i + 1, j)
  ▪ Take the j-th number: (i, j) → (i, j − 1)
Example 2: Algorithm

◊ Taking the $i$-th number:
  - Optimal score for the next player at state $(i + 1, j)$ is $f(i + 1, j)$
  - So the player at state $(i, j)$ will gain $\text{sum}(i, j) - f(i + 1, j)$

◊ Taking the $j$th number:
  - Similarly, will gain $\text{sum}(i, j) - f(i, j - 1)$

◊ $f(i, j) = \max(\text{sum}(i, j) - f(i + 1, j), \text{sum}(i, j) - f(i, j - 1))$

◊ $f(i, j) = \text{sum}(i, j) - \min(f(i + 1, j), f(i, j - 1))$

◊ The final answer: $O(n^2)$
  - $f(1, n)$
  - $\text{sum}(1, n) - f(1, n)$
Outline

- What is combinatorial game?
- Example 1: Simple Game
- Zero-Sum Game and Minimax Algorithms
- **Nim Game**
- Recommended Readings
Nim Game: Setting

◊ Settings:
  - $n$ piles (heaps) of stones.
  - Two players take turns.
  - Each turn, the player chooses a pile, and removes any positive number of stones from the pile.
  - The one who takes the last stones wins.

◊ Goal:
  - Find out the winner if both play optimally
Nim Game: State & Move

◊ State
  ▪ The number of stones in all piles
  ▪ $O(m^n)$ state space, where $m$ is the maximum number of stones in a single pile

◊ We can’t really use DP since the state space will be huge for large number of piles
Nim Game: Example

◊ Starts with heaps of 3, 4, 5 stones
  ▪ Call them heap A, B, and C respectively
◊ Player 1 takes 2 stones from A: (1, 4, 5)
◊ Player 2 takes 4 from C: (1, 4, 1)
◊ Player 1 takes 4 from B: (1, 0, 1)
◊ Player 2 takes 1 from A: (0, 0, 1)
◊ Player 1 takes 1 from C and wins: (0, 0, 0)
Nim Game: Algorithm

◊ Given heaps of size $n_1, n_2, \ldots, n_m$

◊ Claim
  - The first player wins if and only if the nim sum, $n_1 \oplus n_2 \oplus \ldots \oplus n_m$ is nonzero (bitwise XOR operation: ^ in C/C++, Java, Python)
Nim Game: Proof

◊ Similar to Example 1: induction!
◊ It holds for the losing state \((0,0,\ldots,0)\) since the nim sum is 0.
◊ If the nim sum is 0, then whatever the current player does, nim sum of the next state is non-zero
  ▪ Because there is only one number changed
◊ If the nim sum is nonzero, it is possible to force it to become 0
  ▪ Not obvious, but true
  ▪ Refer to Wikipedia for more details
  ▪ “Proof of the winning formula” in https://en.wikipedia.org/wiki/Nim
Outline

◇ What is combinatorial game?
◇ Example 1: Simple Game
◇ Zero-Sum Game and Minimax Algorithms
◇ Nim Game
◇ Recommended Readings
Recommended Readings

- Sprague–Grundy theorem
- Variations of Nim