Computing Science (CMPUT) 657

Algorithms for Combinatorial Games

Martin Müller

Department of Computing Science
University of Alberta
mmueller@ualberta.ca

Winter 2022

Algorithms for Impartial Games

Computational Issues for Impartial Games

- Goal: review impartial games in terms of computation, including MEX rule, and work by Lemoine and Viennot
- Impartial Games - both players have the same options throughout
- Classic example: Nim
- Can modify games to make them impartial
  - Example: Cram = Domineering where both players can play horizontal or vertical
- General idea: impartial version $I(G)$ of game G:
  - Each player gets all options of both players from the partizan version
  - $I(G) = \{I(G^{\mathcal{L}}), I(G^{\mathcal{R}}) | I(G^{\mathcal{L}}), I(G^{\mathcal{R}})\}$
- In MCGS and CGSuite: impartial wrapper

Impartial Games Quick Review

- Outcome classes $\mathcal{N}$ and $\mathcal{P}$ only, cannot be $\mathcal{L}$ or $\mathcal{R}$
- $G = -G$, game is its own inverse
- Sprague-Grundy theorem: each finite impartial game is equal to some Nim heap $*n$, $n \geq 0$
- *0 = 0, *1 = *,
  $*n = \{*0, *1, ... *(n-1) | *0, *1, ... *(n-1)\}$
- *0 is the only $\mathcal{P}$-position
- All other $*n$ are $\mathcal{N}$-positions, win by moving to 0

Impartial Games Review - Nim Addition

- Nim addition: to compute $*a + *b$
  - Write a and b as sums of powers of 2, cancel equal powers
  - Example: $*5 + *7 = *4 + *1 + *4 + *2 + *1 = *2$
  - Equivalent: Write a and b in binary, compute bitwise XOR. *101 + *111 = *(1 XOR 1)(0 XOR 1)(1 XOR 1) = *010 = *10 in binary

Mex - the Minimum EXcluded Number

- Definition
- S = finite set of nonnegative integers
- $mex(S)$ = Smallest integer $n \geq 0$ which is not in set S
- Examples
  - $mex(\{0, 1, 2\}) = 3$
  - $mex(\{0, 1, 4, 5\}) = 2$
  - $mex(\{1, 2, 3\}) = 0$
  - Exercise: $mex(\{0, 5, 1, 3, 7, 2, 6\}) = ?$
  - Exercise: $mex(\{3, 2\}) = ?$

Mex Rule - Computing a Game From its Options

- Given $G = \{G^{\mathcal{L}} | G^{\mathcal{R}}\}$
- Since G is impartial, $G^{\mathcal{L}} = G^{\mathcal{R}}$
- Assume (through recursive evaluation) that all games in $G^{\mathcal{L}}$ have been simplified to nim heaps
- $G = \{*n_1, *n_2, ..., *n_k | *n_1, *n_2, ..., *n_k\}$
- Let $n = mex(\{n_1, n_2, ..., n_k)\}$.
- Mex is the *minimum excluded number*
- Theorem (e.g. Siegel, Theorem 1.2): $G = *n$

Examples

- $G = \{*1, *3, *0 | *1, *3, *0\}$
- $mex(\{1, 3, 0\}) = 2$
- $G = *2$
- $H = \{*1, *3, *2 | *1, *3, *2\}$
- $H = 0$
- In class: compute nim-values for `q1game` for board sizes 1..10

Proof of Mex Theorem

- $G = \{*n_1, *n_2, ..., *n_k | *n_1, *n_2, ..., *n_k\}$
- Let $n = mex(\{n_1, n_2, ..., n_k)\}$. Theorem: $G = *n$
- We show that $G - *n$ = 0, a second player win
- Case 1:
  - Player 1 moves to some $*k$ with $k < n$, in either G or $*n$.
  - Then player 2 can copy that move, leaving $*k + *k = 0$
- Case 2:
  - Player 1 moves in G, to some $*k$ with $k > n$.
  - Then player 2 can move from $*k$ to $*n$, leaving $*n + *n = 0$

More Mex Examples

- Example $G = \{*1, *3, *0 | *1, *3, *0\} = *2$
- We show that $G + *2 = 0$
- Example: move G to $*1$:
  Win *1+*2 by moving to *1+*1
- Example: move G to $*3$:
  Win *3+*2 by moving to *2+*2
- Example: move $*2$ to $*1$:
  Win G+*1 by moving to *1+*1
- Exercise: work out the winning ways for the other moves

Computation, “Linear strip” Games

Clarification: what I meant here are games where each position can be described by a single parameter n.
This does NOT include games such as linear Clobber or linear NoGo, where there are exponentially many different games on a size n board.

- Computation can be much simpler than for partizan games!
- A game $*n$ can be represented by an integer n
- Cost: linear in size of game tree in practice
- Bottleneck: compute mex. $O(1)$ per option if Nim values are bounded (e.g. bitvector)
- Large nim values are very rare in practice
- For some games played on a $1 \times n$ strip, billions of Nim-values have been computed
- Typical goal: show that they eventually become *periodic*

Paper: Nimbers are Inevitable

- J. Lemoine and S. Viennot, Nimbers are inevitable (2012) - see readings
- Breakthrough algorithm for solving complex impartial games
- Applied to Sprouts and Cram (impartial Domineering)
- Greatly increased the number of results known for these games

Nimbers are Inevitable - Overview

- First half of paper is review of impartial theory and negamax alphabeta
  - Unfortunately they use non-standard notation for everything...
- Second half of paper is crystal clear and beautiful
- Main result: while solving for win/loss of an impartial sum game $G + H$, you can get the nimber value of at least one subgame (either G or H) for free
- Turned this observation into an efficient algorithm by solving sums $G + *n$

Basic Facts (in Standard Notation)

- If $G \in \mathcal{P}$ and $H \in \mathcal{P}$ then $G + H \in \mathcal{P}$
- Proof: $G \in \mathcal{P} \Leftrightarrow G = 0$ and $0 + 0 = 0$
- If $G \in \mathcal{P}$ and $H \in \mathcal{N}$ then $G + H \in \mathcal{N}$
- Proof: $G + H = 0 + H = H$
- Case: $G \in \mathcal{N}$ and $H \in \mathcal{N}$
  - $G + H \in \mathcal{P}$ iff G and H are equal to the same nimber $*n$

Theorem 3 in Lemoine and Viennot

- Assume we have an algorithm to solve $G + H$ by game tree search and store the proof tree
- Then, with no extra search, we can determine the nim value of at least one of G or H
- Proof by Induction on $G + H$ - we assume it is true for the options from $G + H$
- Base case: no options in either subgame: $G = H = *0$
- Recursive cases on next slide

Theorem 3 Proof Continued

Case 1: $G + H$ was a win, $G + H \in \mathcal{N}$

- There exists winning move, for example from G to some $G^{\mathcal{L}}$
- (WLOG: all other cases are analogous, e.g. win in H, or Right's move)
- Resulting position is loss (for opponent), so $G^{\mathcal{L}} + H = 0$
- By induction assumption, we know the nim value of either or $G^{\mathcal{L}}$ or H
- Since their sum is 0, they must be the same, so we know some n such that $G^{\mathcal{L}} = H = *n$.
- Since we know that $H = *n$, we know the nim value of one subgame in $G + H$

Theorem 3 Proof Continued

Case 2: $G + H$ is a loss, $G + H \in \mathcal{P}$

- Any move leads to a win for the opponent
- If we know the nim value of H, there is nothing left to prove
- So look at the case where we do not know the nim value of H
- WLOG consider all Left options in G, $G_1, ...G_k$
- By induction assumption, we know the nim value of all $G_1, ...G_k$ (since we know it for one of G_i, H for each i, and we do not know it for H)
- We can use the mex rule to compute nim value of G!

Algorithm

- Compute outcomes of $G = G + *0$, $G + *1$,...$G + *n$
- Stop when we hit a loss, then $G = *n$
- Store all in a single table
- Moves in $G + *n$: move in G, or move in $*n$ if $n > 0$
- Boolean negamax: return win as soon as a move leads to a loss for the opponent
- Return loss if all moves lead to a win (for them)
- Can do either:
  - Bottom-up computation (database, retrograde analysis) or
  - Top-down computation (cache and re-use all results in a transposition table)

Sum Algorithm

- If position G splits into subgames $G_1 + G_2 + ...G_k$
- To compute win/loss result for $G + *n$
- Pick one special subgame G_k, for example the largest/hardest one
- Compute all other nim values separately for $G_1, \ldots, G_{k-1}$
  - Collect results: $G_1 = *n_1, \ldots, G_{k-1} = *n_{k-1}$
- Compute nim sum of $*n$ plus these other subgames, $*n' = *n + *n_1 + *n_2 + ... * n_{k-1}$
- Now search $G_k + *n'$. It is a win if and only if $G + *n$ is a win
  - Do you see why?

Algorithm Engineering and Comments

- Choice of which subgame G_k to search last, with one fixed nim heap $*n'$ only
- Move ordering for the boolean negamax
- Comments
  - The idea is a special case of solving $G + \mathit{inf}$ for simple infinitesimals
  - Boolean win/loss only, thermographs do not make sense (they all look the same except for $n = 0$)
  - Can use any boolean solver such as boolean negamax, proof-number search, df-pn

Applications - Solving Impartial Games

Sprouts

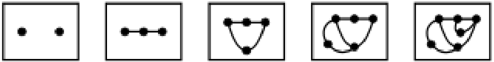

FIGURE 1. Example of a Sprouts game, starting with 2 spots (the second player wins).

- Paper and pencil game
- Start with a number of dots
- Move: connect two dots with a line, and add a new dot in the middle
- Constraints
  - Lines cannot cross
  - Dots cannot be used in more than 3 lines

Sprouts Results

- Several Lemoine and Viennot papers, some focus on game-specific decomposition rules. Current records - see references page
- Computed all Sprouts starting positions with up to $p = 44$ dots, some positions up to $p = 53$ dots
- Previous state of the art: $p = 14$ (!) Huge advance.
- First approach to really use subgame decomposition for Sprouts
- Works really well, subgames appear early in search
- Sprouts conjecture: game is a loss ($\mathcal{P}$-position) iff $p \equiv 0, 1, 2 \pmod 6$

Cram Rules and Symmetry

- Cram = Domineering where both players can place a domino in any way - either horizontal or vertical
- Cram on rectangular boards:
- Even by even is 0 by simple mirror strategy
  - Always copy opponent's move mirrored from center
- Even by odd is first player win by simple mirror strategy
  - Occupy two center squares on first move
  - Then mirror-copy all other moves
  - These are the "not *0" entries on next page

Cram Results - Nim Values

- Values up to $3 \times 20, 4 \times 9, 5 \times 9, 6 \times 7, 7 \times 7$ boards
- No apparent structure
- $3 \times n$, $n = 1..20$: *1, *1, *0, *1, *1, *4, *1, *3, *1, *2, *0, *1, *2, *3, *1, *4, *0, *1, *0, *2
- $4 \times n$, $n = 4..9$: *0, *2, *0, *3, *0, *1
- $5 \times n$, $n = 5..9$: *0, *2, *1, *1, *1
- $6 \times n$, $n = 6..9$: *0, *5, *0, not *0
- $7 \times n$, $n = 7..8$: *1, not *0
- Works less well than for sprouts, subgames appear later in search - similar to Amazons?

Possible Course Projects and Readings

- MCGS impartial game wrapper - can study impartial version of any game
- Do you have an impartial game that you want to solve?
- We have an "almost finished" version of this algorithm
- There is also the Beling and Rogalski algorithm that claims to be more efficient
  - A good paper to present in class
  - Their source code is available (but hard to understand for me)
  - Re-implement it in MCGS?

Summary

- With Nim addition and Mex rule, we can efficiently compute values of impartial games
- Many results for "one-dimensional" games played with numbers or strips, e.g. subtraction games
- Theorem 3 in Lemoine and Viennot was a breakthrough result for solving "two-dimensional", general impartial games efficiently
- Pushed analysis of Sprouts and Cram much further