Competitive Programming

From Problem 2 Solution in O(1)

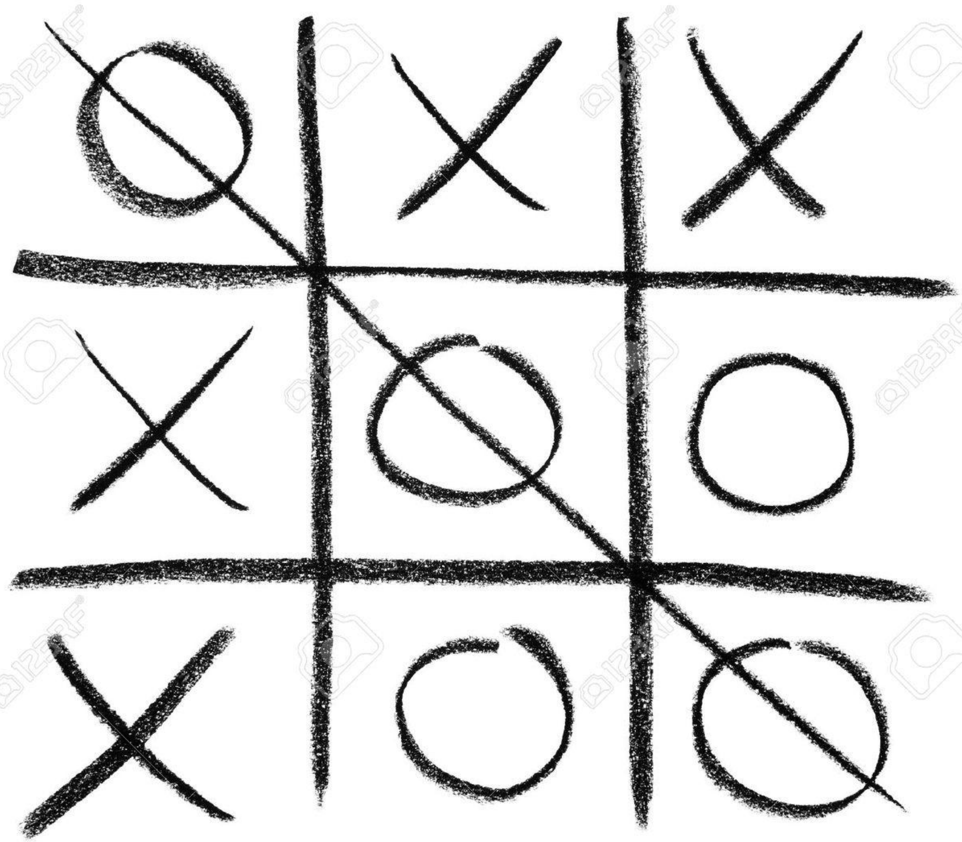
Combinatorial Game Theory

Introduction

Mostafa Saad Ibrahim
PhD Student @ Simon Fraser University

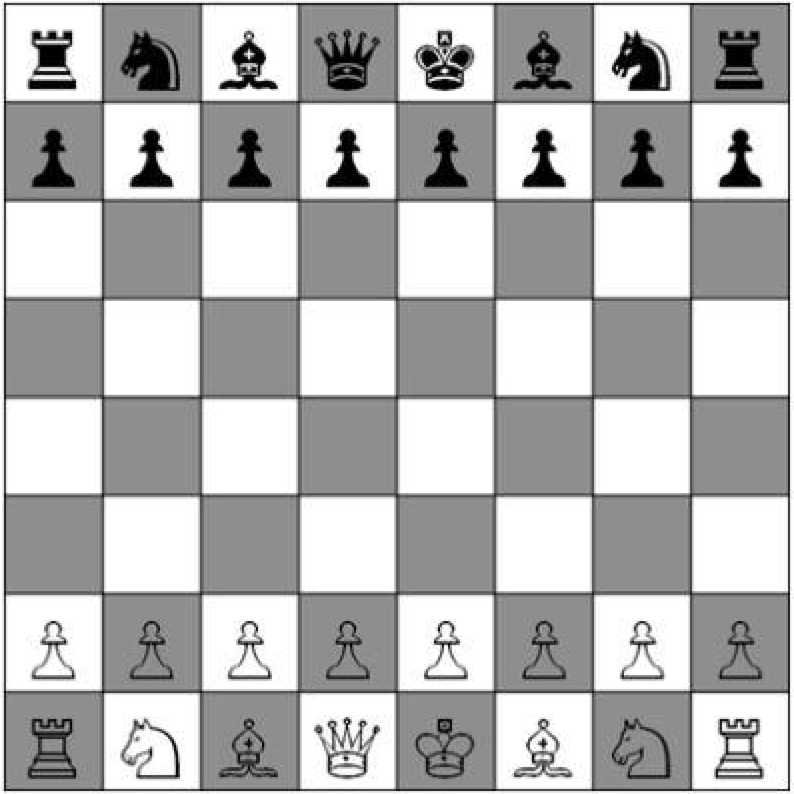
Tic-Tac-Toe Game

- **Two players** (turn by turn)
- All information available for all
    - **Perfect information**
- One allowed to put X only
- Other allowed to put O only
- Different moves = **Partisan**
- **Finite steps** (9 steps)
- **Combinatorial game**



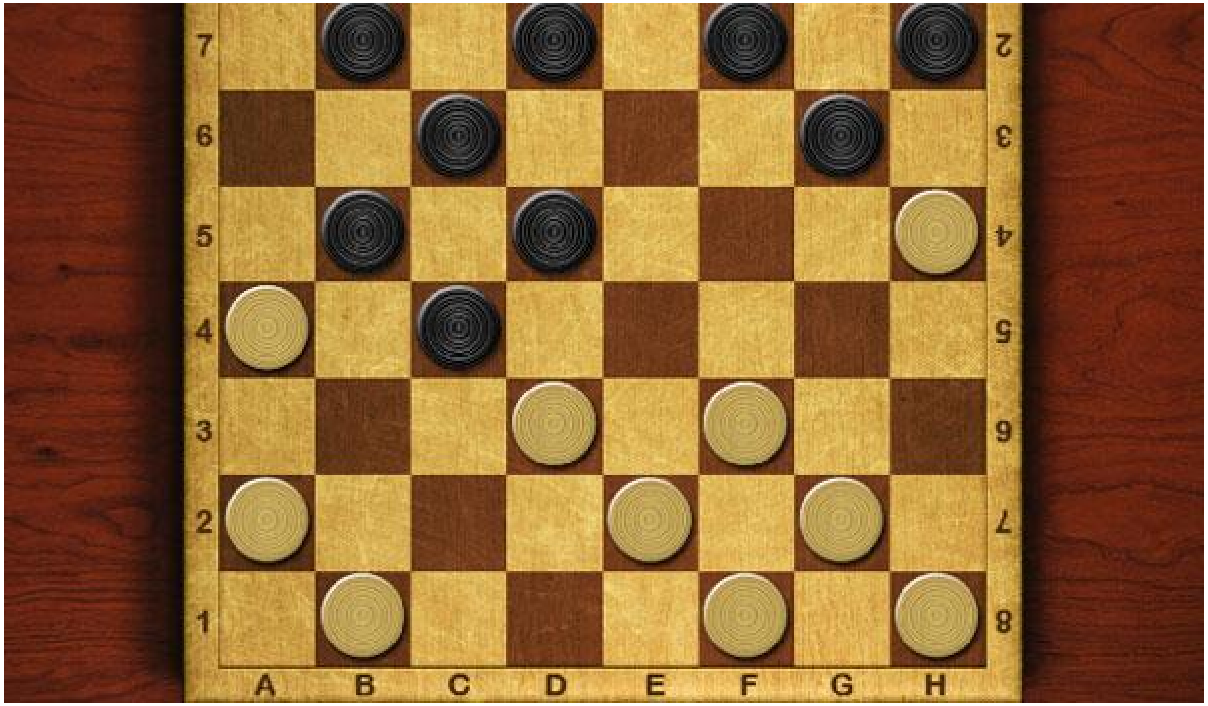
Src: http://crystalclearfinances.com/wp-content/uploads/2016/06/3526260-Hand-drawn-tic-tac-toe-game-isolated-on-white-Stock-Photo.jpg

Chess Game



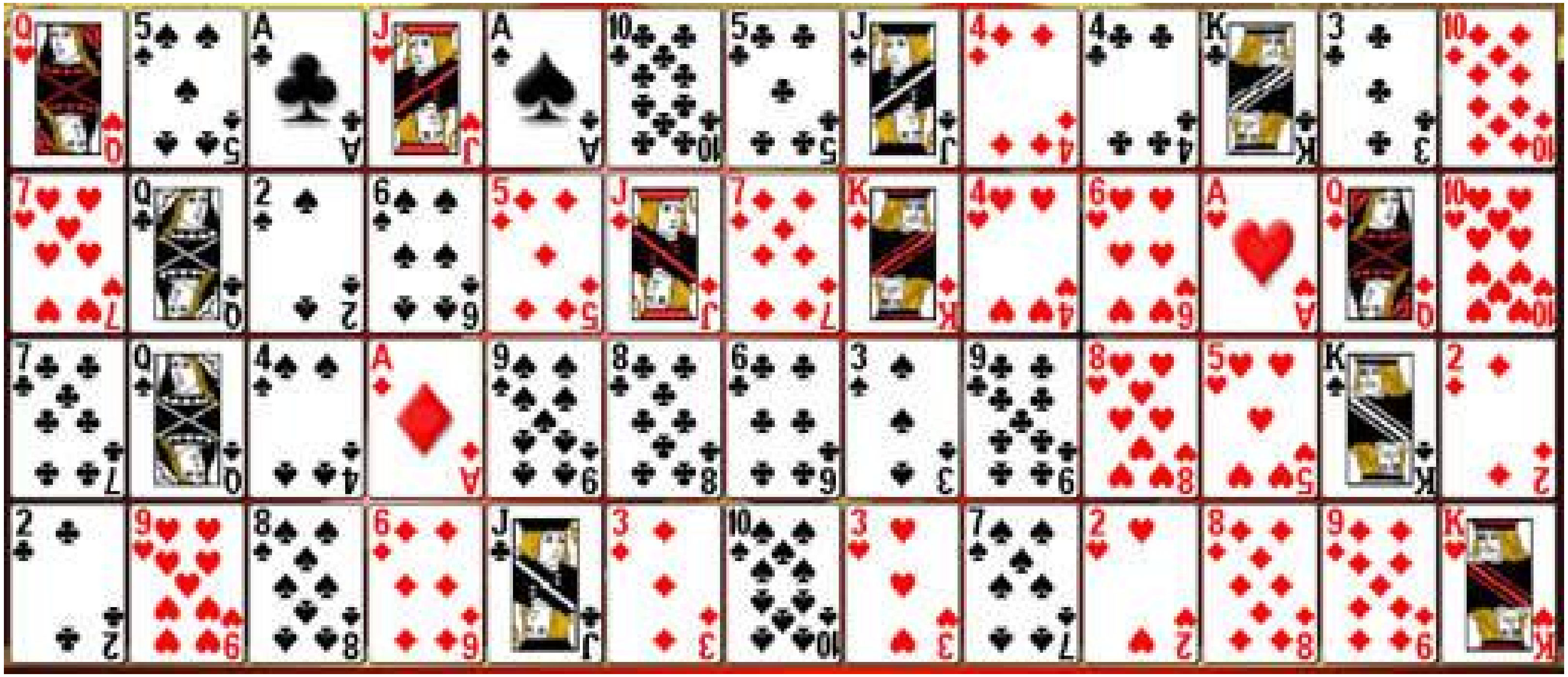
Src: http://www.activityvillage.co.uk/sites/default/files/images/chess-board-layout.jpg

Checkers Game



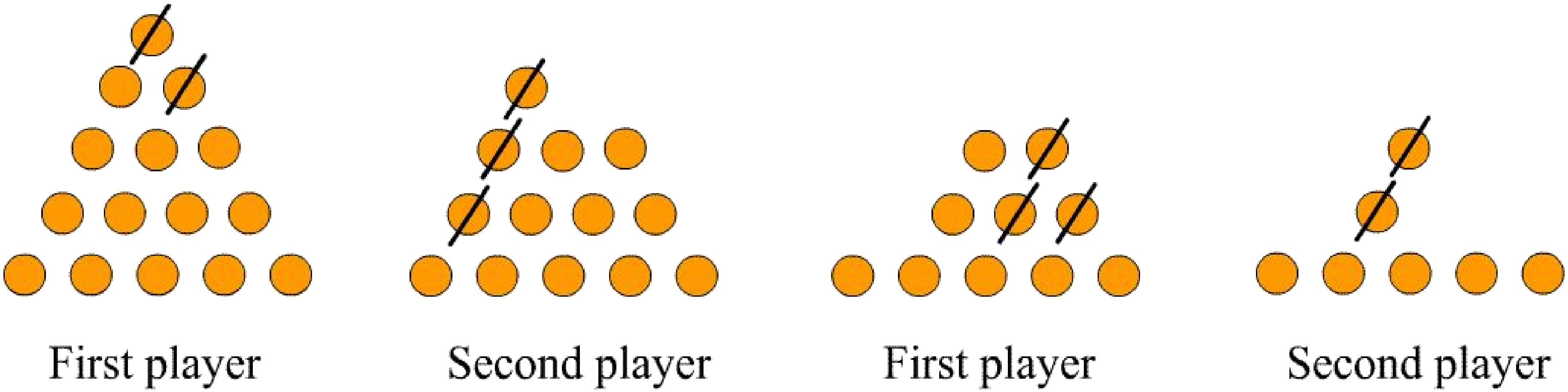
Src: http://www.coolmath-games.com/sites/cmatgame/files/checkers.jpg

52-card deck Game



Src: http://www.solitairenetwork.com/images/r70.jpg

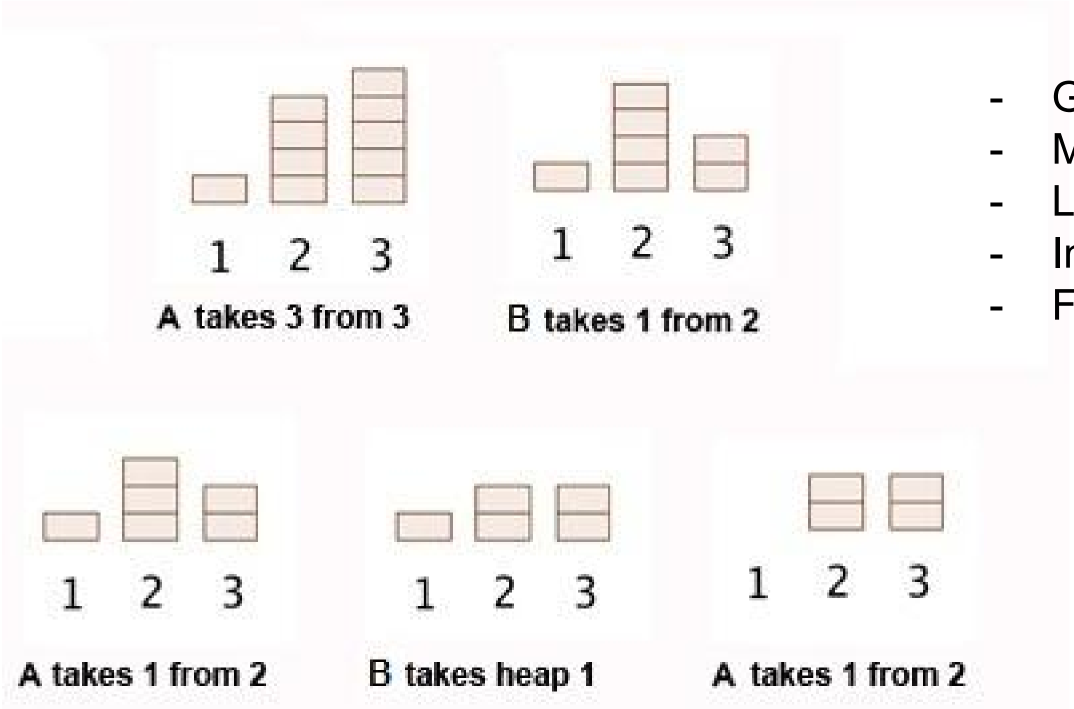
Game of Nim (1 pile of stones)



- Each turn, pick 1 or more items. Winner is player to remove the last items
- Input: pile of 15. Moves: remove 2, 3, 3, 2 ...

Src: http://www.sfcc.edu.hk/academic_subjects/mathematics/web/1999_2000_projects/Clara%20To/Nim2.gif

Game of Nim (many piles)



- Given multiple piles of stones
- Move: take any stones from ONE pile
- Loser: Nothing to take
- Input: (1, 4, 5)
- First move: (1, 4, 2) = 3 from pile # 3

Src: http://www.geeksforgeeks.org/combinatorial-game-theory-set-2-game-nim/

Go Game

Src: http://paulomenin.github.io/go-presentation/images/goban.png

Backgammon Game

Randomization

Src: https://upload.wikimedia.org/wikipedia/commons/3/30/Backgammon_lg.png

Rock Paper Scissors

Parallel moves

Src: http://blog.heartland.org/wp-content/uploads/2011/03/rockpaperscissors.jpg

Jenga

Src: https://images.vat19.com/covers/large/giant-jenga.jpg

Game Theory

- It is used in economics, political science, and psychology, **computer science**, ...etc
- Many types / properties
  - **Two players** vs Many players
  - **Sequential** vs Simultaneous (e.g. Rock Paper Scissors)
  - **Perfect information** vs imperfect information
  - **Finite games** vs Infinitely long games
  - **Discrete** vs continuous games
  - **Combinatorial games** (Partisan vs **Impartial** Games)
- Our focus: See **bold** words above

Combinatorial Games (of interest)

- Two players + moving alternately
  - Select first player. A plays, then B, then A, then B ...
- The winner is determined by who moves last.
  - Or in misère variation who can't move last
- Perfect information
  - All state information is available (as in chess), no hidden information (as in most of card games)
- Finite game, No draws, No randomization
- Two types: Partisan vs **Impartial Games**

Games Examples

- Combinatorial Games
  - **Game of Nim (most important for us)**
  - Chess, Checker (hard)
    - Many ad hoc problems related to them
  - Tic-Tac-Toe (trivial), Go (complex: go vs chess)
- Non Combinatorial Games
  - 1 player: Tower of Hanoi, Sudoku [disagreements]
  - Backgammon: **Dice** = Randomization
  - 52-card deck: **Imperfect** information (you don't know what in opponent hands)

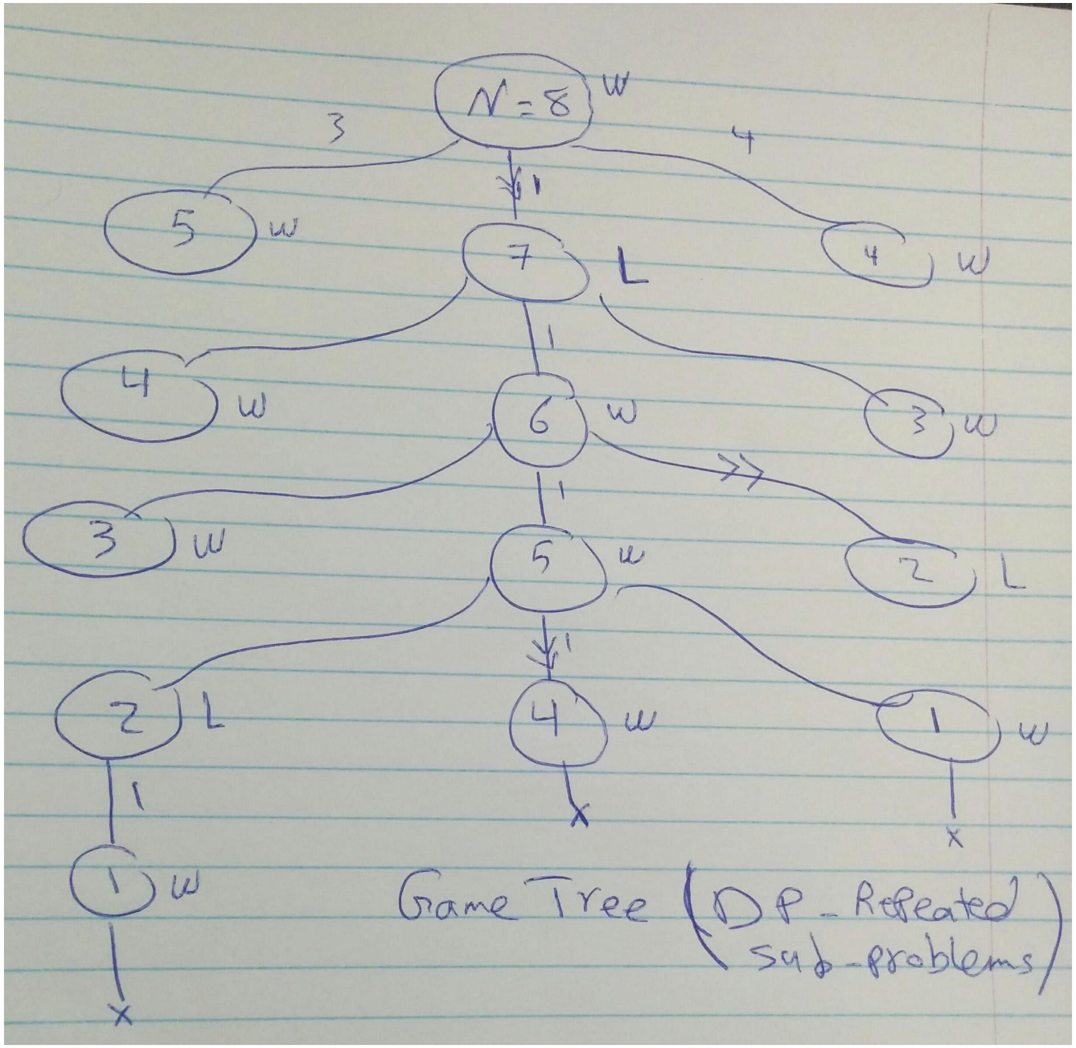
Partisan vs Impartial Games

- Two major categories of Combinatorial games
- **Impartial** Games
  - **Same** set of **moves** at any time allowed for both players
  - **NIM**: For any player, pick whatever items (same moves)
  - **Jenga**
- **Partisan** Games
  - The moves for all the players are not the same.
  - Tricky and hard to analyze
  - **Chess**: Each player moves only his own pieces
  - **Tic-Tac-Toe**: I add X, but you add O (also allow draws)
  - Typically backtracking/dp/minimax coding

Let’s play: Nim game - 1 pile

- Given N stones in a game of 2 players
  - In each move, the player can withdraw 1, 3 or 4 stones
  - The last player to take stones is the winner
  - Assume players **play optimally**
- **Winning** position
  - Any position where a player can make a move and play optimally in all next move and **will win**
- **Losing** position
  - Whatever player trials, he will lose at the game end
- **Terminal** position: No further possible moves
  - Winner/loser are declared

Game Tree



N=8
W
3
1
4
5
W
7
L
4
W
4
W
1
3
W
6
W
1
3
W
5
W
2
L
1
2
L
4
W
1
W
1
1
W
x
x
x
Game Tree (DP - Repeated sub-problems)

Recursive Win/Lose code

```
bool isWinning(int pos) {
  if (pos == 0)
    return false;  // can't move = terminal position

  int moves[3] = { 1, 3, 4 };
  // play optimally: try all against his optimality
  for (int i = 0; i < 3; ++i) {
    if (pos >= moves[i] && !isWinning(pos - moves[i]))
      // opponent will lose from this move
      return true;  // ANY lose = I win
  }
  return false;  // ALL moves make opponent win
}
```

Code notes

- Typically, we use memoization (DP)
- Like fibonacci, we can write 1 loop to build such solution (Table method)
- Being 0/1 output with dependency on last k entries, this sequence will be **periodic** (e.g. don't need compute to N)
  - Compute cycle, precycle
  - Identify where N will be in them

Solution table

n	0	1	2	3	4	5	6	7	8	9	10	11
position	L	W	L	W	W	W	W	L	W	L	W	W

- Initialize Base case from n = 0 to n = 4
- Build incrementally
  - N = 8 ⇒ Check in table: A[7] A[5] A[4] => {L, W, W} = W
  - N = 9 ⇒ Check in table: A[8] A[6] A[5] => {W, W, W} = L

Src: https://www.topcoder.com/community/data-science/data-science-tutorials/algorithm-games/

Position properties

- Winning position
  - From current position move to **any** losing position
  - Then opponent lose in this posiion $\Rightarrow$ hence you win
- Losing position
  - From current position, **all** moves are winning positions
  - Then opponent will always win $\Rightarrow$ hence you lose
- Terminal position
  - Losing position = Base Cases

Your turn

- 1) **Misère** invariant: **Loser** is **last** one to move
  - Compute the table as we did
- 2) what if we are allowed to take only 1 or 2 stones?
  - Can you get a **simple winning strategy** to determine the answer (trivial processing)?
  - Yes, if n-1 or n-2 is divisible by 3, first player wins
    - Just a simple rule instead of recursive code
  - Compute the table as we did and verify

Move duplication strategy

- Given 2 piles of sizes x > 0, y > 0
  - User can take any amount stones
  - If x == y case
    - **Second** player can always **win** by **duplicating** what 1st player do
    - For example if given (9, 9). Player one took 5, then you take 5. He took 4, then you take 4. He loses
  - If x != y
    - **First** will **win**. Let your first step to make it x = y.
    - E.g. for (2, 5) => takes 3 from 2nd pile => (2, 2)
    - Whatever 2nd move (1 or 2), he loses finally as 1st case above for (x == y)

Move duplication strategy

- Sometimes we **duplicate** in a **mirroring** way
  - So we create a symmetric behaviour
  - Find some point to create to make symmetry/reflections around
  - Think in a 2D grid or 1D grid
  - Each time user mark/take from the grid, we mirror it
  - Overall strategy end as if we are **cancelling** user actions
  - We will see example soon

Move cancellation strategy

- Given 2 piles of sizes x > 0, y > 0
  - User can take any amount stones
  - User can also **add stones**, but **not exceeding** the **original** number of stones in that pile
  - **Intuition**: The adding move is useless! It is normal nim
  - Any player can always **cancel** *adding stones move*
  - E.g. if user add 5 stones, just remove them. Hence game went back to its **old status**. So better don't do such move.
  - This is called **Poker Nim**
  - Note, the **duplication** strategy in 2 pile is kind of **cancellation** strategy: e.g.
    - *every 2 equal piles are cancelled*

Example: Bowling Pins

- Game
  - N bowling pins arranged in a row
  - Move 1: Hit exactly one pin (See image 1 - top part)
  - Move 2: Hit in middle of any two adjacent pins
  - Hint: Use Duplication with mirroring strategy
  - Hint: Mirror around the middle of the row

- Remove 2nd pin

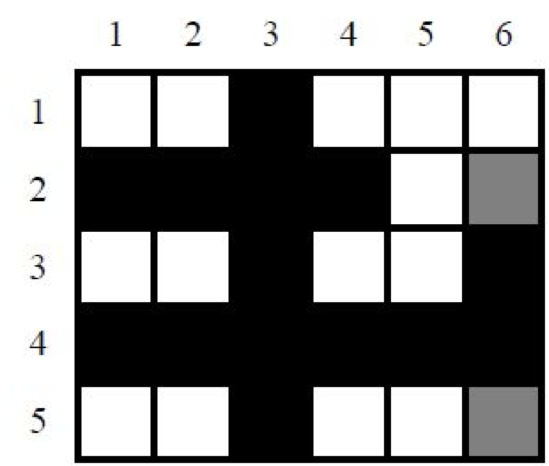
- Remove 5th, 6th bins

Example: Bowling Pins

- Game
  - Let N be odd
  - Let your first hit to remove middle one
  - Now we have N/2 on left and same on right
  - **Mirror user action**
  - E.g. if removed 2 from left => remove 2 from right
  - E.g. if removed 1 from right => remove 1 from left
  - Then you must remove last pins and win :)
  - What N was even? Remove middle 2 oines

Your turn: Black out problem

- Black Out Problem
  - Matrix 5 x 6 of white cells.
  - Move: color adjacent cells (in a row or column) as black
    - Some of them might be labeled already, but not all
  - Loser: Nothing remain to color
  - Think in a winning strategy
  - Hint: Use **duplication with mirroring strategy**
  - Solution in next slide



Solution: Black out problem

- Notice we have even # of cells
  - In your first action, mark whole 3rd row, remains 24 cells
  - Follow a mirroring strategy (duplicate actions that make grid symmetric relative to 3rd row)
  - E.g. if 2nd player marked top 4 cells in first column, then mark the bottom 4 cells in first column (you will mark some cells that already marked)
  - If user marked whole row, then mark another complete row (in symmetry style)
  - Note, without symmetry (e.g. just coloring same # of new cells), user might end with 2 cells in different rows/cols such that he has to mark 1 end, and then other win!

Your turn: Coin in grid

- Given a 2D rectangle N x M. There is a coin in position (N, M)
- Move: Either move to decrease its row only or column only
  - E.g. given (10, 20) can move to (10, 17) or (5, 20)
  - But you can’t decrease both: e.g. (5, 17)
- Loser: Can’t move
- Map it to a Nim game and solve it
- See solution in next slide

Solution: Coin in grid

- It is a 2 pile game.
- Pile 1 = N, Pile 2 = M
- Then solve 2 piles name as mentioned before

Your turn

- Given 1 pile of N items. Given M <= N
- Move: take any value from 1 to M
- Prove that You will lose IFF
  - N % (M+1) == 0
  - See
- Later we shall see easy way to solve that

Your turn

- Given 2 piles of stones with moves:
- Move 1: Select a pile and take 1 stone
- Move 2: Take 1 stone from both piles
- Move 3: Move 1 stone from pile to another
- Under normal rules, who win?

Your turn

- Read the Ioiwari Game (IOI 2001)
  - Don't solve :)
  - Just determine:
  - **Is** it solvable using some search technique (e.g. DP)?
  - **Or** we should find some winning strategy?
  - Solution in ppt comments section

Final notes

- Our **math** focus will be on **Impartial** Games (specially Nim game)
  - Specifically, finding a winning strategy
- Other game types appear in contests, but typically involve **search** techniques with **pruning** (Chess, Checker, Sudoku, Hanoi, Card Games) or **Dynamic** Programming
  - Be aware of standard games rules / terminologies
  - In problem statement, they may alter some rules
  - My dynamic programming playlist provides examples

تم بحمد الله

علمكم الله ما ينفعكم

ونفعكم بما تعلمتم

وزادكم علماً