Ad Hoc

Lecture 3: Summer of Competitive Programming

Soumil Aggarwal

Algorithms and Coding Club
Indian Institute of Technology Delhi

3 July 2021

Ad Hoc - Introduction

- Ad hoc problems are those whose algorithms do not fall into standard categories with well-studied solutions. Each ad hoc problem is different; no specific or general techniques exist to solve them.
- Of course, the above definition is kinda vague, and we're gonna contradict it further by showing you two techniques to solve 'ad hoc' problems.
- Ad hoc problems usually rely more on pure problem solving ability and less on the knowledge of DSA. Trying out cases, making observations, and good problem-solving habits in general are key.

Pigeonhole Principle

It is an intriguing name given to following obvious but very useful fact:

Pigeonhole Principle

If you have $(n+1)$ objects and n boxes to put them in, some box will have ≥ 2 objects.

We also have:

Generalized Pigeonhole Principle

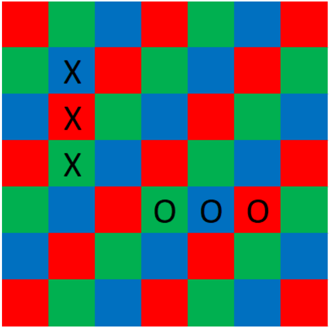
If you have N objects and k boxes to put them in, some box will have $\geq \left\lceil \frac{N}{k} \right\rceil$ objects, and some box will have $\leq \left\lfloor \frac{N}{k} \right\rfloor$ objects

Errich-Tac-Toe Solution

Background - Colouring

- This problem uses the very common idea of coloring a grid according to value of co-ordinates modulo m - If you have colours $c_0, \ldots, c_{m-1}$, then you colour the cell (i, j) with the colour $c_{(i+j)\%m}$, that is the $((i+j)\%m)^{th}$ colour.
- The nice thing about such a colouring is that any m consecutive cells in a row or a column must contain each colour exactly once
- A very common example of this is the chessboard coloring - You colour cells with odd sum of co-ordinates black, and cells with even sum of co-ordinates white.

Errich-Tac-Toe Solution



- In the current problem, we colour the grid modulo 3. To make it more concrete - Colour the cell (i, j) -

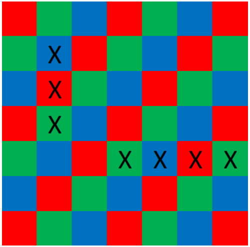
$$\begin{cases} \text{Green} & \text{if } (i+j)\%3 = 0 \\ \text{Blue} & \text{if } (i+j)\%3 = 1 \\ \text{Red} & \text{if } (i+j)\%3 = 2 \end{cases}$$

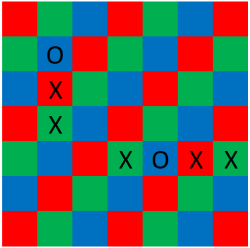
Errich-Tac-Toe Solution

- This is motivated by the **fact** that for any winning configuration, three tokens which cause it to be winning lie on cells of different colours.
- Idea - What if choose a colour and change all the Xs lying on cells of that color to Os.
- Then the configuration will have to be a draw configuration, due to **fact**.

Errich-Tac-Toe Solution

- But, can we do this with $\leq \left\lfloor \frac{k}{3} \right\rfloor$ operations? This is where generalized PHP comes to the rescue! (In a way, use of PHP was motivated by the presence of the floor function.)
- The three colours R, G, B, are the boxes, and the k Xs we're given initially are the objects. By generalized PHP, atleast one of them contains $\leq \left\lfloor \frac{k}{3} \right\rfloor$ Xs. Just change all of them to Os, and we're done.
- The time complexity of the solution is $O(n^2)$, which should easily pass for the given constraints.


X
X
X
X X X X


O
X
X
X O X X