ECE374 SP23 HW6

Contributors

Zhirong Chen (zhirong4)

Ziyuan Chen (ziyuanc3)

Problem 2

Suppose you are given n poker chips stacked in two stacks, where the edges of all chips can be seen. Each chip is one of three colors. A turn consists of choosing a color and removing all chips of that color from the tops of the stacks.

The goal is to minimize the number of turns until the chips are gone. Give an $O(n^2)$ dynamic programming algorithm to find the best strategy for a given pair of chip piles.

Solution

This is a classic backtracking algorithm that resembles searching on a tree with max depth n. At the i^{th} level, the BackTrack function is called for i times. Total time complexity is $O(n^2)$.

```
$\text{ClearStacks}(S_1, S_2)$
    $\text{Path} \leftarrow [\,]$
    $\text{OptimalPath} \leftarrow [\,]$
    $\text{OptimalPathLen} \leftarrow \infty$
    $\text{Backtrack}(S_1, S_2, \text{Path})$
    $\textbf{return}$ $\text{OptimalPath}$

$\text{Backtrack}(S_1, S_2, \text{Path})$
    $\textbf{if}$ $\text{IsEmpty}(S_1)$ $\textbf{and}$ $\text{IsEmpty}(S_2)$
        $\textbf{if}$ $\text{Length}(\text{Path}) < \text{OptimalPathLen}$
            $\text{OptimalPath} \leftarrow \text{Path}$
            $\text{OptimalPathLen} \leftarrow \text{Length}(\text{Path})$
        $\textbf{return}$
        $\textbf{for}$ $\text{Color}$ $\textbf{in}$ $(S_1.\text{top}, S_2.\text{top})$
            $\text{Path.append}(\text{Color})$
            $\textbf{while not}$ $\text{IsEmpty}(S_1)$ $\textbf{and}$ $S_1.\text{top} = \text{Color}$
                $S_1.\text{pop}()$
            $\textbf{while not}$ $\text{IsEmpty}(S_2)$ $\textbf{and}$ $S_2.\text{top} = \text{Color}$
                $S_2.\text{pop}()$
            $\text{Backtrack}(S_1, S_2, \text{Path})$
            $\text{Path.pop}()$
        $\textbf{return}$
```