Theory Assignment-2: ADA Winter-2024

Vivaswan Nawani (2021217) Animesh Pareek (2021131)

1 Subproblem Definition

- We need only one subproblem definition.
- Max_Chickens(i, j) is the maximum number of chickens that Mr. Fox can collect after completing the ith Obstacle and having a count of j. Here, j represents the count of consecutive Rings or Dings used at this point.
- $j = 0, 1, 2$ denotes that the count of Rings is 1, 2, 3 respectively (count of consecutive Rings = $j + 1$).
- $j = 3, 4, 5$ denotes that the count of Dings is 1, 2, 3 respectively (count of consecutive Dings = $(j + 1)/2$).

2 Recurrence of the subproblem

ASSUMPTION: The length of the problem array (Array A) is at least 1.

Base Cases:

$$Max\_Chickens(0, j) = \begin{cases} A[0] & \text{if } j = 0 \\ (-1) \times A[0] & \text{if } j = 3 \\ -\infty & \text{if } j = 1 \text{ or } j = 2 \text{ or } j = 4 \text{ or } j = 5 \end{cases}$$

Also, please note that

$$Max\_Chickens(1, j) = -\infty \text{ if } j = 2 \text{ or } j = 3$$

Recurrence case: For any index $1 < i < |A|$ and $-1 < j < 6$ where $i, j \in N$, we have the following recurrence relation:

$$\texttt{Max\_Chickens}(i, j) = \begin{cases} \max(\texttt{Max\_Chickens}(i-1, 3), \texttt{Max\_Chickens}(i-1, 4), \texttt{Max\_Chickens}(i-1, 5)) + A[i] & \text{if } j = 0 \\ \texttt{Max\_Chickens}(i-1, j-1) + A[i] & \text{if } j = 1 \text{ or } j = 2 \\ \max(\texttt{Max\_Chickens}(i-1, 0), \texttt{Max\_Chickens}(i-1, 1), \texttt{Max\_Chickens}(i-1, 2)) - A[i] & \text{if } j = 3 \\ \texttt{Max\_Chickens}(i-1, j-1) - A[i] & \text{if } j = 4 \text{ or } j = 5 \end{cases}$$

3 The specific subproblem(s) that solves the actual problem

- We need to consider a maximum of 6 subproblems to solve the actual problem.
- Subproblem 1: Max_Chickens($i, 0$) represents the maximum possible chickens collected at the ith index when the number of consecutive RING calls is 1.
- Subproblem 2: Max_Chickens($i, 1$) represents the maximum possible chickens collected at the ith index when the number of consecutive RING calls is 2.
- Subproblem 3: Max_Chickens($i, 2$) represents the maximum possible chickens collected at the ith index when the number of consecutive RING calls is 3.
- Subproblem 4: Max_Chickens($i, 3$) represents the maximum possible chickens collected at the ith index when the number of consecutive DING calls is 1.

- Subproblem 5: Max_Chickens$(i, 4)$ represents the maximum possible chickens collected at the ith index when the number of consecutive DING calls is 2.
- Subproblem 6: Max_Chickens$(i, 5)$ represents the maximum possible chickens collected at the ith index when the number of consecutive DING calls is 3.
- Let Solve(n) represent the problem given in this question, where we need to find the maximum possible chickens that Mr. Fox can collect after covering n obstacles. Then,

$$\text{Solve}(n) = \max_{0 \le j \le 5} \text{Max\_Chickens}(n, j)$$

4 Algorithm description

- Input:
  - n, the number of obstacles
  - $A[i]$ $0 \le i \le n-1$, which store the number of chickens in each obstacle.
- Let $DP[i][j]$, $0 \le i \le n-1$ and $0 \le j \le 5$, store the max possible number of chickens after the ith obstacle and a count of j.
- Initialize $DP[i][j] = -\infty$, for $0 \le i \le n-1$ and $0 \le j \le 5$.
- Let's define a function `Max_Chicken(i, j)`, which takes i and j as input parameters and returns the max possible chickens for this subproblem.
- Max(i, j) will return values based on the following rules:
  - If $i = 0$ and $j = 1, 2, 4, 5$, return $-\infty$
  - Else If $i = 1$ and $j = 2, 5$, return $-\infty$
  - Else If $i = 0$ and $j = 0$, return $A[i]$
  - Else If $i = 0$ and $j = 3$, return $-A[i]$
  - Else If $DP[i][j] \ne -\infty$, return $DP[i][j]$
  - Else
    * If $j = 0$, $DP[i][j] = \max(\texttt{Max\_Chickens}(i-1, 3), \texttt{Max\_Chickens}(i-1, 4), \texttt{Max\_Chickens}(i-1, 5)) + A[i]$
    * Else If $j = 1$ or $j = 2$, $DP[i][j] = \texttt{Max\_Chickens}(i-1, j-1) + A[i]$
    * Else If $j = 3$, $DP[i][j] = \max(\texttt{Max\_Chickens}(i-1, 0), \texttt{Max\_Chickens}(i-1, 1), \texttt{Max\_Chickens}(i-1, 2)) - A[i]$
    * Else If $j = 4$ or $j = 5$, $DP[i][j] = \texttt{Max\_Chickens}(i-1, j-1) - A[i]$
    * return $DP[i][j]$
- Output $\max_{0 \le j \le 5}$ Max_Chickens(n, j)

5 Complexity Analysis

Let $T(A)$ denote the time complexity of the algorithm for the input array A of length n.

- **Initialization**: Initializing the DP array takes $O(6n) = O(n)$ time.
- **Recurrence Relation Evaluation**: Evaluating the recurrence relation involves computing each entry of the DP array. For each entry $DP[i][j]$, we make a constant number of comparisons and possibly recursive calls.
  - Each entry computation involves comparing and selecting the maximum of at most three values (for $j = 0$ or $j = 3$) or two values (for other j).

  - The recursive calls are made for indices $i - 1$, which results in a total of n recursive calls for each j value.

  Therefore, the time complexity of evaluating the recurrence relation is $O(6n) = O(n)$.

- **Base Cases**: Setting up the base cases takes constant time as they involve simple conditional assignments, $O(1)$.
- **Main Loop**: The main loop iterates over the possible values of j once, which takes $O(6) = O(1)$ time.

Therefore, the overall time complexity $T(A)$ of the algorithm is:

$$T(A) = O(n) + O(n) = O(n)$$

$$T(A) = O(n)$$