Algorithms I

Tutorial 5

August 26, 2016

Problem 1

$dp[i]$: The length of the longest monotonically increasing subsequence ending at i
$dp[i] = max(1, max\{dp[j] + 1 \ \mid j < i, A_j < A_i\})$
The answer is maximum among $dp[i]$ for $1 \leq i \leq n$.

Problem 2

$dp[i][0]$: The length of the longest good subsequence of the form $a_1 < a_2 > a_3 .. < a_i$ ending at i
$dp[i][1]$: The length of the longest good subsequence of the form $a_1 < a_2 > a_3 .. > a_i$ ending at i

$dp[i][0] = max(1, max\{dp[j][1] + 1 \ \mid j < i, A_j < A_i\})$
$dp[i][1] = max(1, max\{dp[j][0] + 1 \ \mid j < i, A_j > A_i\})$
The answer is maximum among $dp[i][0], dp[i][1]$ for $1 \leq i \leq n$.

Problem 3

$dp[i][j]$: The probability of obtaining exactly k heads if first i coins are tossed.

- Base case:
$$dp[0][j] = \begin{cases} 1, & \text{if } j = 0 \\ 0, & \text{otherwise} \end{cases}$$
- For $1 \leq i \leq n, 0 \leq j \leq k$:
$$dp[i][j] = \begin{cases} dp[i-1][j] \cdot (1 - p_i), & \text{if } j = 0 \\ dp[i-1][j-1] \cdot p_i + dp[i-1][j] \cdot (1 - p_i), & \text{otherwise} \end{cases}$$

Problem 4

Let us sort the intervals with respect to the right end. So, now we have $r_i < r_j$ for $i <= j$
$dp[i]$: The maximum number of non-overlapping intervals among $(l_1, r_1), (l_2, r_2), \ldots (l_i, r_i)$ (in the sorted set of intervals)

- Base case, $dp[1] = 1$
- for $2 \leq i \leq n$, let j be the maximum index such that $r_j \leq l_i$. If no such j exists, $dp[i] = dp[i-1]$, otherwise $dp[i] = max(dp[i-1], dp[j] + 1)$.

The maximum size of non-overlapping set of intervals is $dp[n]$. This gives $O(n^2)$ algorithm if we search for j using linear search. This can be optimized to $O(n \log n)$ by using binary search to find such j.

Problem 5

$dp[i][j]$: Whether there exist a non empty subset S of $a_1, a_2, \ldots a_i$ such that the sum of elements in $S \equiv j (\text{mod } k)$. 1 if there is a subset, 0 otherwise.

- Base case:
  for $0 \le j < k,\ dp[1][j] = \begin{cases} 1, & \text{if } j = a_i \\ 0, & \text{otherwise} \end{cases}$
- for $2 \le i \le n, 0 \le j < k$
  $dp[i][j] = \begin{cases} 1, & \text{if } j = a_i \\ max(dp[i-1][j], dp[i-1][(j - a_i) \text{ mod } k], & \text{otherwise} \end{cases}$

If $dp[n][0]$ is 1, then there exist a non-empty subset such that sum of elements is divisible by k, otherwise no such subset exists.

Problem 6

$dp[i][j]$: The length of the longest subsequence in the sub-string $S_i S_{i+1} \ldots S_j$.

- Base cases
  - Length 1: $dp[i][i] = 1$, for $1 \le i \le n$
  - Length 2: $dp[i][i+1] = \begin{cases} 2, & \text{if } i+1 \le n, S_i = S_{i+1} \\ 0, & \text{otherwise} \end{cases}$
- For $j - i > 1$:
  $dp[i][j] = \begin{cases} dp[i+1][j-1] + 2, & \text{if } S_i = S_j \\ max(dp[i+1][j], dp[i][j-1]), & \text{otherwise} \end{cases}$

The answer is $dp[1][n]$.

Problem 7

$dp[i]$: 1 if the player starting first can guarantee his/her win, 0 otherwise

- Base case: $dp[0] = 0$
- for $1 \le i \le n$, $dp[i] = 1 - min(dp[i-j], 1 \le j \le i)$
  If there are i stones, the player can remove $1 \le j \le i$ stones. If for any $j, dp[i-j] = 0$, this means the player can remove j stones and the other player can be forced to lose as $i - j$ is a losing position.

Alice can guarantee her win if and only if $dp[n] = 1$.

Problem 8

$dp[i][j]$: The maximum number of coins we can get if we start from $(1, 1)$ and reach (i, j).

- Base case:
  $dp[1][1] = a_{11}$
  for $2 \le j \le m, dp[1][j] = dp[1][j-1] + a_{ij}$
  for $2 \le i \le n, dp[i][1] = dp[i-1][1] + a_{ij}$

- For $2 \leq i \leq n, 2 \leq j \leq m$
  $dp[i][j] = max(dp[i-1][j], dp[i][j-1]) + a_{ij}$

The answer is $dp[n][m]$