Advanced Computer Contest Preparation

Lecture 17

INTERVAL DP

Sample Problem: Coins in a Row

- There are N $(1 \leq N \leq 1{,}000)$ coins in a straight line
- The i^{th} coin has a value of v_i
- You play this game with a friend:
  - Alternating turns, starting with you, remove one coin from one end of the line
  - Add that coin's value to your score
- Your friend plays perfectly
  - Plays in a way to maximize his score
- What is your maximum possible score?

Sample Problem: Coins in a Row

Example game:

4, 4, 9, 4

4, 9, 4 (you get 4)

9, 4 (friend gets 4)

4 (you get 9)

(friend gets 4)

Maximum score is 13

Solutions?

- Greedy?
  - What are possible greedy criteria?
  - $O(N \log N)$ greedy is possible, but algorithm is complex
- Brute force?
  - At each possible turn (there are N turns), there are 2 possibilities
  - Runtime is $O(2^N)$
- DP?
  - What are the subproblems?
  - How does a problem relate to its subproblems?

Interval DP

- A sub-type of the DP technique
- Overall problem is the answer to the entire range, $[1,N]$
- Subproblems are answers to all possible ranges, $[l,r]$, $l \leq r$
- Also called left-right DP or L-R DP

Coins in a Row – DP Solution

- The overall problem is:
  - What is the maximum score if all coins from 1 to N are present, and we go first?
- The subproblems are:
  - What is the maximum score we can achieve if the only coins present are those in the range $[l,r]$, $l \leq r$, and we go first?

DP Solution

- Let $p(l,r)$ be the maximum score we can achieve if coins in the range $[l,r]$ are the only ones present, and we go first
- Base case: $p(l,r) = v_l$ if $l = r$
  - There is only one coin; we simply take that coin

DP Solution

- Case 1: We take coin l
  - Coins in the range $[l+1,r]$ are left
  - Since friend plays perfectly, he gains points equal to $p(l+1,r)$
  - We get all other coins, so our score will be $\sum_{i=l}^{r} v_i - p(l+1,r)$
- Case 2: We take coin r
  - Coins in the range $[l,r-1]$ are left
  - Friend gains $p(l,r-1)$ points
  - Our score is $\sum_{i=l}^{r} v_i - p(l,r-1)$
- We choose which case gives us a better score
- Therefore, $p(l,r) = \sum_{i=l}^{r} v_i - \min(p(l+1,r), p(l,r-1))$

Example 1

Coins: 4, 4, 9, 4

$l \setminus r$	1	2	3	4
1	4	4	13	13
2	–	4	9	8
3	–	–	9	9
4	–	–	–	4

Example 2

Coins: 4, 5, 12, 3, 4, 2

$l \setminus r$	1	2	3	4	5	6
1	4	5	16	16	12	20
2	–	5	12	8	16	10
3	–	–	12	12	15	16
4	–	–	–	3	4	5
5	–	–	–	–	4	4
6	–	–	–	–	–	2

Pseudocode – Recursive

```
int solve(int l, int r){
        if (dp[l][r]) return dp[l][r];
        if (l == r) dp[l][r] = sum[r]-sum[l-1];
        else dp[l][r] = sum[r]-sum[l-1]-min(solve(l+1,r),solve(l,r-1));
        return dp[l][r];
}

print(solve(1,N));
```

Pseudocode – Iterative

```
for (int size = 0; size < N; size++){
        for (int l = 1; l+size <= N; l++){
                int r = l + size;
                if (l == r) dp[l][r] = sum[r]-sum[l-1];
                else dp[l][r] = sum[r]-sum[l-1]-min(dp[l+1][r],dp[l][r-1]);
        }
}

print(dp[1][N]);
```

Analysis

- How many states are there?
  - $O(N^2)$
- How many subproblems does each state depend on?
  - $O(1)$
- What is the time complexity needed to compute the solution to a problem?
  - $O(1)$
- Therefore, the final time complexity is $O(N^2)$

THANK YOU!