ALGORITHM DESIGN PROJECT

Haiao ji 6138009
Ha Ngoc Bao Linh 6138310

Maximum Earnings From Taxi (from leetcode)

Table of Contents

1. The problem

There are `n` points on a road you are driving your taxi on. The `n` points on the road are labeled from `1` to `n` in the direction you are going, and you want to drive from point `1` to point `n` to make money by picking up passengers. You cannot change the direction of the taxi.

The passengers are represented by a **0-indexed** 2D integer array `rides`, where `rides[i] = [start_i, end_i, tip_i]` denotes the ith passenger requesting a ride from point $start_i$ to point end_i who is willing to give a tip_i dollar tip.

For **each** passenger `i` you pick up, you **earn** $end_i - start_i + tip_i$ dollars. You may only drive **at most one** passenger at a time.

Given `n` and `rides`, return *the **maximum** number of dollars you can earn by picking up the passengers optimally.*

Note: You may drop off a passenger and pick up a different passenger at the same point.

Constraints:

- $1 <= n <= 10^5$
- $1 <= rides.length <= 3 * 10^4$
- `rides[i].length == 3`
- $1 <= start_i < end_i <= n$
- $1 <= tip_i <= 10^5$

2. Example

Example 1:

```
Input: n = 5, rides = [[2,5,4],[1,5,1]]
Output: 7
Explanation: We can pick up passenger 0 to earn 5 - 2 + 4 = 7 dollars.
```

Example 2:

```
Input: n = 20, rides = [[1,6,1],[3,10,2],[10,12,3],[11,12,2],[12,15,2],[13,18,1]]
Output: 20
Explanation: We will pick up the following passengers:
- Drive passenger 1 from point 3 to point 10 for a profit of 10 - 3 + 2 = 9 dollars.
- Drive passenger 2 from point 10 to point 12 for a profit of 12 - 10 + 3 = 5 dollars.
- Drive passenger 5 from point 13 to point 18 for a profit of 18 - 13 + 1 = 6 dollars.
We earn 9 + 5 + 6 = 20 dollars in total.
```

3. Analysis

For each ride, we can:

1. Select the ride, get the benefit end - start + tip, and then jump to the end position.

2. Do not select the ride, do not get any benefits, and then go directly to the next location.

Therefore, we can define dp[i] to represent the maximum benefit that can be obtained from location i. Then, traverse all sites from back to front. At each site, if:

1. If there is no ride, you can't get any benefits. dp[i] = dp[i + 1].

2. If there is a ride, you can not select it, and it is still dp[i] = dp[i + 1]. You can also try to select the ride, then dp[i] = dp[ride [end]] + ride [end] - i + ride [tip]. Finally dp[i] choose the larger one.

4. Code

```python
class Solution:
    def maxTaxiEarnings(self, n: int, rides: List[List[int]]) -> int:
        rides.sort()
        dp = [0] * (n+1)
        j = len(rides) - 1

        for i in range(n-1, -1, -1):
            dp[i] = dp[i+1]
            while j >= 0 and rides[j][0] == i:
                s, e, t = rides[j]
                dp[i] = max(dp[i], e-s+t+dp[e])
                j -= 1

        return dp[1]
```

5. Solution

Your input	5 [[2,5,4],[1,5,1]]
Output	7
Expected	7

Your input	20 [[1,6,1],[3,10,2],[10,12,3],[11,12,2],[12,15,2],[13,18,1]]
Output	20
Expected	20

6. Time complexity

Best case	**Worst case**
O(n)	O(nlogn)

7. Submission

Time Submitted	Status	Runtime	Memory	Language
09/22/2021 17:52	Accepted	1760 ms	33.7 MB	python3

8. Reference

https://leetcode.com/problems/maximum-earnings-from-taxi/

https://www.youtube.com/watch?v=OWEG9X-AreY

https://stackoverflow.com/questions/4433915/why-is-sorting-a-string-on-log-n

THANK YOU !