Articles > 1029. Two City Scheduling

Previous (/articles/minimum-number-of-arrows-to-burst-balloons/) Next (/articles/product-of-array-except-self/)

1029. Two City Scheduling (/problems/two-city-scheduling/)

May 10, 2019 | 12.4K views

Average Rating: 4.48 (29 votes)

There are `2N` people a company is planning to interview. The cost of flying the `i`-th person to city `A` is `costs[i][0]`, and the cost of flying the `i`-th person to city `B` is `costs[i][1]`.

Return the minimum cost to fly every person to a city such that exactly `N` people arrive in each city.

Example 1:

```
Input: [[10,20],[30,200],[400,50],[30,20]]
Output: 110
Explanation:
The first person goes to city A for a cost of 10.
The second person goes to city A for a cost of 30.
The third person goes to city B for a cost of 50.
The fourth person goes to city B for a cost of 20.

The total minimum cost is 10 + 30 + 50 + 20 = 110 to have half the people interv
```

Note:

1. `1 <= costs.length <= 100`
2. It is guaranteed that `costs.length` is even.
3. `1 <= costs[i][0], costs[i][1] <= 1000`

☰ Articles > 1029. Two City Scheduling ▾

Solution

Approach 1: Greedy.

Greedy algorithms

Greedy problems usually look like "Find minimum number of *something* to do *something*" or "Find maximum number of *something* to fit in *some conditions*", and typically propose an unsorted input.

> The idea of greedy algorithm is to pick the *locally* optimal move at each step, that will lead to the *globally* optimal solution.

The standard solution has $\mathcal{O}(N \log N)$ time complexity and consists of two parts:

- Figure out how to sort the input data ($\mathcal{O}(N \log N)$ time). That could be done directly by a sorting or indirectly by a heap usage. Typically sort is better than the heap usage because of gain in space.
- Parse the sorted input to have a solution ($\mathcal{O}(N)$ time).

Please notice that in case of well-sorted input one doesn't need the first part and the greedy solution could have $\mathcal{O}(N)$ time complexity, here is an example (https://leetcode.com/articles/gas-station/).

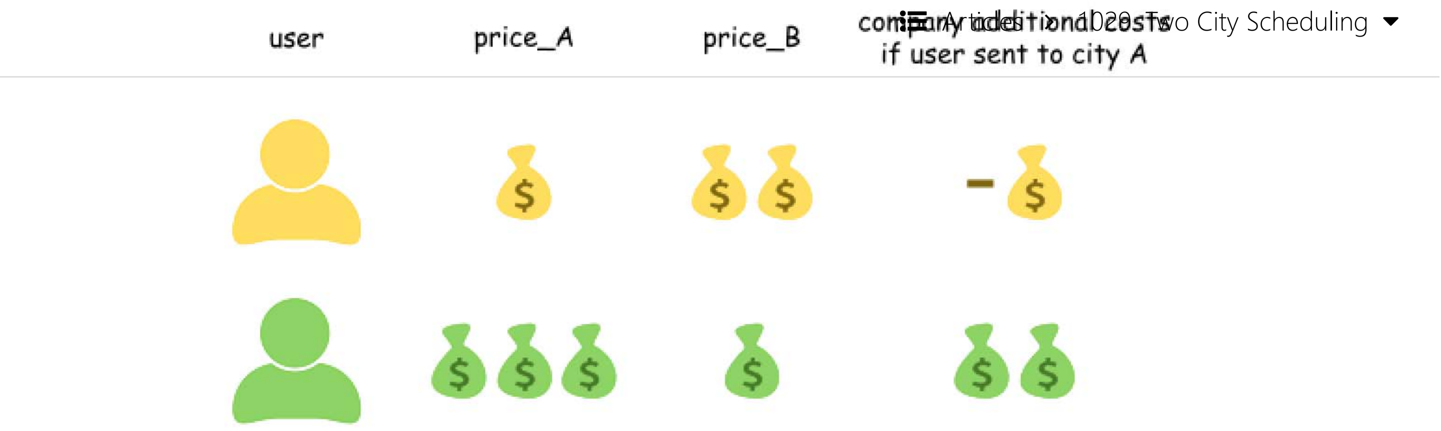
> How to prove that your greedy algorithm provides globally optimal solution?

Usually you could use the proof by contradiction (https://en.wikipedia.org/wiki/Proof_by_contradiction).

Intuition

Let's figure out how to sort the input here. The input should be sorted by a parameter which indicates a money lost for the company.

The company would pay anyway : `price_A` to send a person to the city A, or `price_B` to send a person to the city B. By sending the person to the city A, the company would lose `price_A - price_B`, which could negative or positive.



To optimize the total costs, let's sort the persons by `price_A - price_B` and then send the first `n` persons to the city A, and the others to the city B, because this way the company costs are minimal.

Algorithm

Now the algorithm is straightforward :

- Sort the persons in the ascending order by `price_A - price_B` parameter, which indicates the company additional costs.
- To minimise the costs, send `n` persons with the smallest `price_A - price_B` to the city A, and the others to the city B.

Implementation

C++ Java Python

Copy

```python
class Solution:
    def twoCitySchedCost(self, costs: List[List[int]]) -> int:
        # Sort by a gain which company has
        # by sending a person to city A and not to city B
        costs.sort(key = lambda x : x[0] - x[1])

        total = 0
        n = len(costs) // 2
        # To optimize the company expenses,
        # send the first n persons to the city A
        # and the others to the city B
        for i in range(n):
            total += costs[i][0] + costs[i + n][1]
        return total
```

Complexity Analysis

- Time complexity : $\mathcal{O}(N \log N)$ because of sorting of input data.
- Space complexity : $\mathcal{O}(1)$ since it's a constant space solution.

Rate this article:

Previous (/articles/minimum-number-of-arrows-to-burst-balloons/)

Next (/articles/product-of-array-except-self/)

Comments: 12

Sort By ▾

Type comment here... (Markdown is supported)

Preview Post

(/shuaig1403)

ShuaiG1403 (/shuaig1403) ★ 111 October 10, 2019 3:25 PM

Why is this problem classified as easy?

110 Share Reply

HIDE 3 REPLIES

(/dilip7)

dilip7 (/dilip7) ★ 2 February 23, 2020 5:38 PM

ikr

0 Share Reply

(/mustafa12)

mustafa12 (/mustafa12) ★ 48 February 13, 2020 3:05 PM

thank u!

0 Share Reply

(/lizlizlizzy)

lizlizlizzy (/lizlizlizzy) ★ 3 November 12, 2019 10:13 AM

lol

3 Share Reply

(/premkumarbilla)

premkumarbilla (/premkumarbilla) ★ 92 November 1, 2019 1:15 AM

Lost half of my hair on this...and it says easyy!!!crap

64 Share Reply

(/sinakr2009)

sinakr2009 (/sinakr2009) ★ 9 December 3, 2019 9:41 PM

Here is another greedy criteria that also works, and initially made much more sense to me. But I guess if you analyze it, it is technically very similar to the solution. Just easier to come up with on the spot (for me at least)

Intuition:

Read More

9 Share Reply

SHOW 2 REPLIES

(/xiayizju)

xiayizju (/xiayizju) ★ 5 May 8, 2020 1:39 AM

Please accept my knees, the concept introduced here is opportunity cost (choose A you give up B) and sort based on that. This exactly fits the goal of this problem.

3 Share Reply

(/eivapub)

eivapub (/eivapub) ★ 1 May 28, 2019 8:47 PM

It is not clear: is there always only 2 cities?
And why not to get min on each person? - so result will be O(1) by memory and O(persons*city)?

1 Share Reply

SHOW 2 REPLIES

☰ Articles › 1029. Two City Scheduling ▾

(/nideesht)

NideeshT (/nideesht) ★ 524 ◷ July 29, 2019 1:04 PM

Hi everyone, I'm making Youtube videos to help me study/review solved problems. Wanted to share if it helps!

https://www.youtube.com/watch?v=OkJ1aHjAQr8 (https://www.youtube.com/watch?v=OkJ1aHjAQr8)

Read More

0 ⌃ ⌄ Share Reply

(/ztztzt8888)

ztztzt8888 (/ztztzt8888) ★ 36 ◷ 14 minutes ago

Java 8 with Lambda

```
    public int twoCitySchedCost(int[][] costs) {
        Arrays.sort(costs, (a, b) -> (a[0] - a[1]) - (b[0] - b[1]));
        int sum = 0;
```

Read More

0 ⌃ ⌄ Share Reply

(/mpwind)

mpwind (/mpwind) ★ 0 ◷ May 20, 2020 10:00 PM

Implement the comparator is much easier than writing the sort functions to sort...

0 ⌃ ⌄ Share Reply

(/bll)

Bll (/bll) ★ 51 ◷ April 23, 2020 6:33 PM

this is kind of similar to submodular optimization idea, instead we minimize the max possible regret over each person.

0 ⌃ ⌄ Share Reply

(/ping_pong)

ping_pong (/ping_pong) ★ 711 ◷ February 11, 2020 10:12 PM

@andvary (https://leetcode.com/andvary) Couldn't understand why the approach works ? Can you please explain in more details.

0 ⌃ ⌄ Share Reply

‹ 1 2 ›

Help Center (/support/) | Terms (/terms/) | Privacy (/privacy/) United States (/region/)