2977. Minimum Cost to Convert String II

Description

You are given two **0-indexed** strings `source` and `target`, both of length `n` and consisting of **lowercase** English characters. You are also given two **0-indexed** string arrays `original` and `changed`, and an integer array `cost`, where `cost[i]` represents the cost of converting the string `original[i]` to the string `changed[i]`.

You start with the string `source`. In one operation, you can pick a **substring** `x` from the string, and change it to `y` at a cost of `z` **if** there exists **any** index `j` such that `cost[j] == z`, `original[j] == x`, and `changed[j] == y`. You are allowed to do **any** number of operations, but any pair of operations must satisfy **either** of these two conditions:

- The substrings picked in the operations are `source[a..b]` and `source[c..d]` with either `b < c` **or** `d < a`. In other words, the indices picked in both operations are **disjoint**.
- The substrings picked in the operations are `source[a..b]` and `source[c..d]` with `a == c` **and** `b == d`. In other words, the indices picked in both operations are **identical**.

Return *the **minimum** cost to convert the string* `source` *to the string* `target` *using **any** number of operations*. *If it is impossible to convert* `source` *to* `target`, *return* `-1`.

Note that there may exist indices `i`, `j` such that `original[j] == original[i]` and `changed[j] == changed[i]`.

Example 1:

```
 Input: source = "abcd", target = "acbe", original = ["a","b","c","c","e","d"], changed = ["b","c","b","e","b","e"], cost = [2,5,5,1,2,20]
 Output: 28
 Explanation: To convert "abcd" to "acbe", do the following operations:
- Change substring source[1..1] from "b" to "c" at a cost of 5.
- Change substring source[2..2] from "c" to "e" at a cost of 1.
- Change substring source[2..2] from "e" to "b" at a cost of 2.
- Change substring source[3..3] from "d" to "e" at a cost of 20.
The total cost incurred is 5 + 1 + 2 + 20 = 28.
It can be shown that this is the minimum possible cost.
```

Example 2:

```
 Input: source = "abcdefgh", target = "acdeeghh", original = ["bcd","fgh","thh"], changed = ["cde","thh","ghh"], cost = [1,3,5]
 Output: 9
 Explanation: To convert "abcdefgh" to "acdeeghh", do the following operations:
- Change substring source[1..3] from "bcd" to "cde" at a cost of 1.
- Change substring source[5..7] from "fgh" to "thh" at a cost of 3. We can do this operation because indices [5,7] are disjoint with indices picked
in the first operation.
- Change substring source[5..7] from "thh" to "ghh" at a cost of 5. We can do this operation because indices [5,7] are disjoint with indices picked
in the first operation, and identical with indices picked in the second operation.
The total cost incurred is 1 + 3 + 5 = 9.
It can be shown that this is the minimum possible cost.
```

Example 3:

```
 Input: source = "abcdefgh", target = "addddddd", original = ["bcd","defgh"], changed = ["ddd","ddddd"], cost = [100,1578]
 Output: -1
 Explanation: It is impossible to convert "abcdefgh" to "addddddd".
If you select substring source[1..3] as the first operation to change "abcdefgh" to "adddefgh", you cannot select substring source[3..7] as the
second operation because it has a common index, 3, with the first operation.
If you select substring source[3..7] as the first operation to change "abcdefgh" to "abcddddd", you cannot select substring source[1..3] as the
second operation because it has a common index, 3, with the first operation.
```

Constraints:

- `1 <= source.length == target.length <= 1000`
- `source`, `target` consist only of lowercase English characters.
- `1 <= cost.length == original.length == changed.length <= 100`
- `1 <= original[i].length == changed[i].length <= source.length`
- `original[i]`, `changed[i]` consist only of lowercase English characters.
- `original[i] != changed[i]`
- `1 <= cost[i] <=` 10^6