6097. Match Substring After Replacement

My Submissions (/contest/biweekly-contest-80/problems/match-substring-after-replacement/submissions/) Back to Contest (/contest/biweekly-contest-80/)

You are given two strings `s` and `sub`. You are also given a 2D character array `mappings` where `mappings[i] = [old_i, new_i]` indicates that you may **replace** any number of `old_i` characters of `sub` with `new_i`. Each character in `sub` **cannot** be replaced more than once.

Return `true` *if it is possible to make* `sub` *a substring of* `s` *by replacing zero or more characters according to* `mappings`. Otherwise, return `false`.

A **substring** is a contiguous non-empty sequence of characters within a string.

User Accepted:	0
User Tried:	0
Total Accepted:	0
Total Submissions:	0
Difficulty:	Hard

Example 1:

```
Input: s = "fool3e7bar", sub = "leet", mappings = [["e","3"],["t","7"],["t","8"]]
Output: true
Explanation: Replace the first 'e' in sub with '3' and 't' in sub with '7'.
Now sub = "l3e7" is a substring of s, so we return true.
```

Example 2:

```
Input: s = "fooleetbar", sub = "f00l", mappings = [["o","0"]]
Output: false
Explanation: The string "f00l" is not a substring of s and no replacements can be made.
Note that we cannot replace '0' with 'o'.
```

Example 3:

```
Input: s = "Fool33tbaR", sub = "leetd", mappings = [["e","3"],["t","7"],["t","8"],["d","b"],["p","b"]]
Output: true
Explanation: Replace the first and second 'e' in sub with '3' and 'd' in sub with 'b'.
Now sub = "l33tb" is a substring of s, so we return true.
```

Constraints:

- `1 <= sub.length <= s.length <= 5000`
- `0 <= mappings.length <= 1000`
- `mappings[i].length == 2`
- `old_i != new_i`
- `s` and `sub` consist of uppercase and lowercase English letters and digits.
- `old_i` and `new_i` are either uppercase or lowercase English letters or digits.

JavaScript

```
let ma;
const matchReplacement = (ss, s, mappings) => {
    let n = ss.length, se = new Set(), q = [s];
    ma = build(mappings);
    for (let i = 0; i < n; i++) {
        for (let j = i; j < n; j++) {
            let e = ss.slice(i, j + 1);
            if (e.length == s.length) se.add(e);
        }
    }
    for (const t of se) {
        if (canMake(s, t)) return true;
    }
    return false;
};

const canMake = (s, t) => { // s -> t
    let n = s.length;
    for (let i = 0; i < n; i++) {
```

```
        if (s[i] == t[i]) continue;
        if (!ma.has(s[i])) return false;
        let a = ma.get(s[i]);
        if (a.indexOf(t[i]) == -1) return false;
    }
    return true;
};

const build = (mappings) => {
    let m = new Map();
    for (const [x, y] of mappings) {
        if (!m.has(x)) m.set(x, []);
        m.get(x).push(y);
    }
    return m;
};
```

Custom Testcase | Use Example Testcases

Run | Submit

Submission Result: Accepted (/submissions/detail/719780178/) | More Details > (/submissions/detail/719780178/)

Share your acceptance!