616. Add Bold Tag in a String

June 10, 2017 | 20.8K views

Average Rating: 2.93 (29 votes)

Given a string **s** and a list of strings **dict**, you need to add a closed pair of bold tag `<b>` and `</b>` to wrap the substrings in s that exist in dict. If two such substrings overlap, you need to wrap them together by only one pair of closed bold tag. Also, if two substrings wrapped by bold tags are consecutive, you need to combine them.

Example 1:

```
Input:
s = "abcxyz123"
dict = ["abc","123"]
Output:
"<b>abc</b>xyz<b>123</b>"
```

Example 2:

```
Input:
s = "aaabbcc"
dict = ["aaa","aab","bc"]
Output:
"<b>aaabbc</b>c"
```

Note:

1. The given dict won't contain duplicates, and its length won't exceed 100.
2. All the strings in input have length in range [1, 1000].

Approach #1: Brute Force [Accepted]

Intuition

Let's try to learn which letters end up bold, since the resulting answer will just be the canonical one - we put bold tags around each group of bold letters.

To do this, we'll check for all occurrences of each word and mark the corresponding letters bold.

Algorithm

Let's work on first setting `mask[i] = true` if and only if the `i`-th letter is bold. For each starting position `i` in `S`, for each `word`, if `S[i]` starts with `word`, we'll set the appropriate letters bold.

Now armed with the correct `mask`, let's try to output the answer. A letter in position `i` is the first bold letter of the group if `mask[i] && (i == 0 || !mask[i-1])`, and is the last bold letter if `mask[i] && (i == N-1 || !mask[i+1])`. Alternatively, we could use `itertools.groupby` in Python.

Once we know which letters are the first and last bold letters of a group, we know where to put the `"<b>"` and `"</b>"` tags.

Java | Python — Copy

```python
class Solution(object):
    def boldWords(self, S, words):
        N = len(S)
        mask = [False] * N
        for i in xrange(N):
            prefix = S[i:]
            for word in words:
                if prefix.startswith(word):
                    for j in xrange(i, min(i+len(word), N)):
                        mask[j] = True

        ans = []
        for incl, grp in itertools.groupby(zip(S, mask), lambda z: z[1]):
            if incl: ans.append("<b>")
            ans.append("".join(z[0] for z in grp))
            if incl: ans.append("</b>")
        return "".join(ans)
```

Complexity Analysis

- Time Complexity: $O(N \sum w_i)$, where N is the length of `S` and w_i is the sum of `W`.
- Space Complexity: $O(N)$.

Analysis written by: @awice.

Rate this article:

Previous Next

Comments: 15

Sort By

Type comment here... (Markdown is supported)

Preview Post

patforna ★ 60 July 19, 2018 3:34 PM

Another way to think about this:

1. create a list of tuples/intervals with opening/closing positions. e.g. (open_index, close_index)
2. merge the list of intervals (see https://leetcode.com/problems/merge-intervals/)
3. go through the merged interval list and insert the tags into the string

Read More

58 Share Reply

SHOW 7 REPLIES

achuthna ★ 15 March 9, 2018 9:34 AM

Where exactly is the second solution? I don't see it now. Is there anything better than brute force?

15 Share Reply

stack_UnderFlow ★ 110 April 18, 2018 11:08 PM

You dont need boolean array and save O(N) space like this:

```
class Solution {
    public String addBoldTag(String s, String[] dict) {
```

Read More

9 Share Reply

SHOW 2 REPLIES

whatsgood ★ 52 December 31, 2018 8:42 AM

The Java solution needs to be updated. the entire match method is never used and there's an int anchor that is also never used.

7 Share Reply

alessandrosolbiati ★ 79 January 14, 2019 3:04 AM

amazing use of groupby!

2 Share Reply

asperas ★ 168 March 12, 2020 6:30 PM

Hahaha, how this problem is not even hard?
Since it could be solved by combinations of "merge intervals" + Trie which is both leetcode medium

1 Share Reply

dongseong ★ 71 October 22, 2018 4:09 AM

99% beat O(n w_n) python. w_n is num of dict

```
class Solution:
    def addBoldTag(self, s, dict):
```

Read More

1 Share Reply

SHOW 1 REPLY

FailingAndTrying ★ 4 March 21, 2018 3:43 AM

why break the outerloop

0 Share Reply

8939123 ★ 28 December 5, 2017 2:00 PM

Can you please explain more on time complexity analysis? The 1st method I thought it is s^2: 2nd solution where does the x come from? I thought it is O(ls). Thanks!

0 Share Reply

berkeleysucks ★ 14 November 13, 2017 3:50 AM

The complexity analysis for method#2 is incorrect. We also need to factor in the runtime for sorting.

0 Share Reply