Back

Short Python

StefanPochmann ★ 47270 Last Edit: October 24, 2018 4:12 AM 7.3K VIEWS

51

Either don't encode `s` at all, or encode it as **one** part `k[...]` or encode it as **multiple** parts (in which case we can somewhere split it into two subproblems). Whatever is shortest. Uses @rsrs3's nice trick of searching `s` in `s + s`.

```
def encode(self, s, memo={}):
    if s not in memo:
        n = len(s)
        i = (s + s).find(s, 1)
        one = "%d[%s]" % (n / i, self.encode(s[:i])) if i < n else s
        multi = [self.encode(s[:i]) + self.encode(s[i:]) for i in xrange(1, n)]
        memo[s] = min([s, one] + multi, key=len)
    return memo[s]
```

Comments: 10 Best Most Votes Newest to Oldest Oldest to Newest

Type comment here... (Markdown is supported)

Post

29834829 ★ 24 Last Edit: September 24, 2018 5:39 AM

Curious how `i = (s + s).find(s, 1)` accomplishes finding the optimal repeating x for a "k[x]" encoding of s. Any explanation of this?

J.K. should've read @rsrs3 's post, nice solution though :)

5 Show 1 reply Reply

wyp70627768 ★ 788 January 12, 2019 7:01 AM

damn

1 Reply

rsc571 ★ 57 December 13, 2018 1:42 PM

great solution! I may have a naive question but do you mean by ...

multi = [self.encode(s[:i], memo) + self.encode(s[i:], memo) for i in xrange(1, n)]

also would you mind explaining the complexity? Thanks

1 Show 2 replies Reply

ghad ★ 36 Last Edit: July 12, 2019 9:04 PM

Awesome Solution!!
Java version:

```
public static String encode1(String s, Map<String, String> map) {
    if (map.containsKey(s)) return map.get(s);
    if (s.length() <= 4) return s; //adding this further reduces the runtime from 168 ms to 134 ms
    int n = s.length();
    int i = (s + s).indexOf(s, 1);
```