Nikolas Mavrogeneiadis
gravitorious
November 30, 2024

Nearly Shortest Repeating Substring

Problem: *Link*
For the solution, check the next page.

Solution one:

Let the input string s with size n. We will check every integer $i \in [1, n)$ starting from 1 that $n\%i == 0$ holds using the following process.

Fix i and let us split the string into segments $\{t_1, t_2, ..., t_{n/i}\}$ (i.e. $t_1 = s[0]+s[1]+...+s[i-1]$ is the first segment, etc.). The string we are searching (if it exists) is obviously one of the segments (because if it isn't and because we have at least two segments, then the result string c has more than one conflict, something that it is impossible). Fix two segments t_x, t_y that have the **most** conflicts (if the most conflicts are zero, then we don't have conflicts and we accept i as the result - conflicts means the number of different characters in the same positions). If they have exactly one conflict, then one of them should be exactly the same with all the other segments, other than t_x, t_y, to be able to accept i as the answer (why?). To check this we can compare the j_{th} - or *each_char* in the code - (for every $j \in [0, i-1]$) character for each pair of segment (t_w, t_{w+1}), $w \in [1, n/i)$ starting for $w = 1$ while being careful which characters we compare after we find the first conflict. When we encounter the **first** conflict between the segments, let's say t_x, t_{x+1}, then we will keep the character from t_x (and not from t_{x+1}) to compare with the character from t_{x+2}. This happens because (from hypotheses - we get the first conflict on t_x, t_{x+1}) the character on the t_x is the same with the characters on the previous segments (we are always talking for the same positions inside the segments - in the outside loop for j_{th} character), so all the other characters from t_{x+2} should be equal to the character from t_x. If the conflict is at the start (means between t_1, t_2), we will check which of them does not have conflict with the t_3 and we will keep this characher. If both of them have conflict with t_3, then the conflcits are at least 2 and we can end the process. In the end we should find at most one conflict. If this is the case, we return i. If we don't find any i with at most one conflict, we return n. The complexity is $i * n$ that is about $10^2 * n$ in this problem.

Solution two:

Let s be the input string with length n, and $c = k + k + ... + k$ for the shortest string k that s and c have the same length and differ at most one position.

If such a k exists, then it is on the prefix or the suffix of s. If it wasn't (and k has size h), then comparing it with $s[0]...s[h-1]$ and $s[n-h+1]...s[n]$ we would find at least two conflicts, something that it is impossible. Then, we can check for every suitable i (starting from the smallest) all the prefixes and suffixes. This will take approximately $10^2 * 2 * n$ operations.