Kick Start 2017 - Round E

Analysis: Copy & Paste

Copy & Paste: Analysis

Small dataset

Let N be the length of the target string **S**. We can observe that:

- Copying and pasting a single character is always strictly worse than just typing another instance of that character, which takes only one operation.
- Copying and pasting two or more characters can be useful, but it turns out not to be for N ≤ 5. Suppose that we wanted to use copying and pasting to create a string of length 5. Then the only possible copy/paste operation that uses two or more characters is to start with a string of length 3 and copy/paste two of its characters. But then we could have just typed the two characters instead, using the same number of operations.
- For N = 6, there are some patterns for which the optimal strategy requires copying and pasting. With some brute-force experimentation or case-based analysis, you can find that they are of all of the of the form 111111, 121212, or 123123 (in which each number consistently stands for a particular letter)..

It is also possible to use a brute-force breadth/depth-first search to solve the Small.

Large dataset

For the Large dataset, we can use dynamic programming. We will keep track of the minimum number of operations used to reach a state with the following parameters, as the target string is being built from left to right:

- Current length - the length of the prefix of the target string that we have built so far
- Clipboard content - the content in the clipboard

There can be at most $O(N^2)$ substrings, so there are $O(N^3)$ states in total. For each state, we can choose to type a character, paste the contents of the clipboard, or copy something into the clipboard. Since there are $O(N^2)$ substrings that we could choose to copy, there are $O(N^2)$ possible moves from each of the $O(N^3)$ possible states, and so the overall time complexity is $O(N^5)$. This approach is fast enough to pass all the test cases within the time limit.

Notice that the copy operation is needed only when the copied string needs to be pasted as the next operation. It reduces the number of candidate strings that need to be copied in each move to O(N) because the number of substring starting with next character is O(N). Thus, this approach leads to an $O(N^4)$ solution.

There is even an $O(N^3)$ approach. Can you find it?