Kick Start 2022 - Round E

Analysis: Matching Palindrome

We should not consider any string Q longer than $\mathbf{P}$ itself because $\mathbf{PP}$ is a palindrome and we can always use $Q = \mathbf{P}$ if there is no shorter valid string Q. Hence, in what follows, we assume that $|Q| \leq |\mathbf{P}|$.

Since $\mathbf{P}Q$ is a palindrome, $\mathbf{P}$ must start with the reverse of Q, which is Q itself since Q is a palindrome. Thus $\mathbf{P} = QX$ for some suffix X.

Test Set 1

We can check every prefix Q of $\mathbf{P}$ and verify if Q and $\mathbf{P}Q$ are both palindromes. The shortest of such valid prefixes Q is our answer. Any given prefix Q can be verified in linear time, and the overall time complexity of this brute-force algorithm is therefore $O(\mathbf{N}^2)$.

Test Set 2

The time complexity of the naive algorithm above can be improved to $O(\mathbf{N})$ using string hashing, which is a widely applicable technique for various string matching problems. In this section, though, we discuss two other approaches.

Manacher's Algorithm

So we are looking for the shortest palindromic prefix Q such that $\mathbf{P} = QX$ and QXQ is also a palindrome. It follows that the suffix X must be palindromic as well. Our task is then to split the string $\mathbf{P}$ into two palindromes Q and X such that Q is as short as possible. Finding such a split would be a trivial task if we knew in advance if any particular prefix or suffix of $\mathbf{P}$ is a palindrome.

This is where the linear time Manacher's algorithm can help. Its main purpose is to find the longest palindromic substring in a given string, however, the algorithm does more than that. Assuming $\mathbf{P} = p_1p_2p_3\ldots p_{\mathbf{N}}$, Manacher's algorithm finds the longest palindromic substring of odd length centered at any particular letter p_i and similarly the longest palindromic substring of even length centered at any particular pair of letters p_ip_{i+1}. Now, how do we know if a certain prefix $\mathbf{P} = p_1p_2p_3\ldots p_k$ of, say, odd length is a palindrome? It is a palindrome precisely when the length of the longest palindromic substring centered at $p_{\frac{k+1}{2}}$ is k. In a similar vein, we can test if a prefix of even length or any suffix is palindromic.

The overall time complexity of this solution is $O(\mathbf{N})$.

Optimized Brute-Force Algorithm

There is a much simpler algorithm if you trust your intuition that for the shortest prefix Q, the string $\mathbf{P}$ must be of the form $\mathbf{P} = Q^k$ for some $k \geq 1$. If this assumption holds true, then we can use a modification of the same brute-force algorithm above, where we test a prefix Q only if $|Q|$ divides $\mathbf{N}$. There are only 128 divisors of $\mathbf{N}$ in the worst case given the constraints of the problem, and that is a small enough constant.

The time complexity of this modified algorithm is $O(\mathbf{N}\sqrt{\mathbf{N}})$ or better depending on the function we use for approximating the number of divisors of $\mathbf{N}$.

It remains to show that our assumption indeed holds.

Claim: If X, Y, and $S = XY$ are all palindromic strings, then $X = Z^k$ and $Y = Z^l$ for some palindrome Z and some integers $k \geq 1$ and $l \geq 1$.

Proof: We will prove the claim using mathematical induction on $|S|$. Without loss of generality, let us assume that $|X| \leq |Y|$. Since S is a palindrome with the prefix X, S must also end in X (and so must Y). Now, since Y is a palindrome with the suffix X, Y must also start with X, and consequently S has a prefix XX. And again, since S is a palindrome, XX must be a suffix of both S and Y. And continuing this back and forth process, we may conclude that $S = X^t W X^t$, where W is a palindromic substring in the middle of S and $|W| < 2|X|$. We consider four possible cases:

- $|W| = \emptyset$: This is a base case of our inductive proof. The entire string S is a power of X, hence we can choose $Z = X$.
- $|W| = |X|$: Another base case. Since the palindrome Y equals $X^{t-1} W X^t$, W must be X, and again, S is a power of X.
- $0 < |W| < |X|$: Since $Y = X^{t-1} W X^t$ is a palindrome, the substring WX in the middle must be palindromic as well. Recall that both W and X are palindromes and $|WX| < |S|$, hence, by inductive hypothesis, W and X must be powers of some palindrome Z, and so is S.
- $|X| < |W| < 2|X|$: Again, $Y = X^{t-1} W X^t$ is a palindrome, so W must be of the form XR, where R is palindromic. Since $|W| < |S|$, the palindromes X and R are powers of some palindrome Z by inductive hypothesis.

That concludes the proof of our claim.

To connect the dots, let us see how this claim validates our optimization of the brute-force algorithm. Suppose we split $\mathbf{P}$ into two palindromes Q and Y and $|Q|$ does not divide $\mathbf{N}$. Then, by the claim, both Q and Y are powers of a shorter palindrome Z, and Q can be disregarded in favour of Z.