Tarea 5: Manacher y Z

Algoritmos Avanzados

David Míreles Gutiérrez A00836010

Monterrey, N.L. 9/29/2024

PseudoCódigo y Análisis de Complejidad

```python
def manacher(longestPalindromeString):  # O(1)
    preprocessedString = '#' + '#'.join(longestPalindromeString) + '#'  # O(n)
    lengthOfString = len(preprocessedString)  # O(1)
    currentCenter = 0  # O(1)
    currentRightBoundary = 0  # O(1)
    palindromeRadii = [0] * lengthOfString  # O(n)

    for idx in range(lengthOfString):  # O(n)
        mirroredIndex = 2 * currentCenter - idx  # O(1)

        if idx < currentRightBoundary:  # O(1)
            palindromeRadii[idx] = min(currentRightBoundary - idx, palindromeRadii[mirroredIndex])  # O(1)

        while (idx + palindromeRadii[idx] + 1 < lengthOfString and  # O(1)
               idx - palindromeRadii[idx] - 1 >= 0 and  # O(1)
               preprocessedString[idx + palindromeRadii[idx] + 1] == preprocessedString[idx - palindromeRadii[idx] - 1]):  # O(1)
            palindromeRadii[idx] += 1  # O(1)

        if idx + palindromeRadii[idx] > currentRightBoundary:  # O(1)
            currentCenter = idx  # O(1)
            currentRightBoundary = idx + palindromeRadii[idx]  # O(1)

    maxLengthPalindrome = max(palindromeRadii)  # O(n)
    centerIndexOfLongestPalindrome = palindromeRadii.index(maxLengthPalindrome)  # O(n)

    startIndex = (centerIndexOfLongestPalindrome - maxLengthPalindrome) // 2  # O(1)

    return longestPalindromeString[startIndex: startIndex + maxLengthPalindrome]  # O(n)
```

La complejidad del algoritmo es **O(n)**, ya que el bucle for principal recorre el texto de longitud n una vez, con una complejidad de O(n). Aunque el bucle while está anidado dentro del for, este solo se ejecuta cuando es necesario expandir el palíndromo, y cada carácter se expande como máximo una vez a lo largo de todo el texto, lo que garantiza que el número total de ejecuciones del while sea también O(n). El segundo bucle for, que encuentra la longitud máxima del palíndromo, tiene igualmente una complejidad de O(n). Lo padre de este

algoritmo es ver que aun teniendo dos loops anidados la complejidad es O(n) dado que el while no se ejecuta en cada iteración por lo que evita la complejidad de O(n^2). Al principio del semestre si veía este código hubiera dicho que la complejidad era O(n^2).