Análisis de Algoritmo

COSTO ALGORÍTMICO

Line	Code	Operations Elementals
1	**public static List<Integer> BoyerMooreHorspool(String subcadena, String texto) {**	
2	List<Integer> posiciones = new ArrayList<>();	1 + 1+ k
3	if (subCadena.length <= 0){	1
4	return posiciones;	1
5	}	
6	int[] tablaDeSaltos = preproceso(subCadena);	1 + 1 + n
7	for (int i = 0; i <= texto.length - subCadena.length;) {	Σ(
8	for (int j = subCadena.length - 1;; --j) {	Σ(
9	if (subCadena.charAt(j) != texto.charAt(i + j)) {	1 + 1 + k
10	int caracterActual = texto.charAt(i + subCadena.length - 1);	1 + 1 + k + 1
11	int valorSalto = (caracterActual < tablaDeSaltos.length) ? tablaDeSaltos[caracterActual] : subCadena.length;	1 + 1 + 1 + k + 1
12	i += valorSalto;	1 + 1
13	break;	1
14	}	
15	if(j == 0) {	1
16	posiciones.add(i);	k
17	i += subCadena.length;	1 + 1
18	break;	1
19	}	
20	}	)
21	}	)
22	return posiciones;	1
23	}	
24	T(n)	3kmn + 13mn + 3m + 3n + k + 9
25	O(n)	**nm**

Análisis de Algoritmo

Universidad Francisco de Paula Santander

Vigilada Mineducación

Programa de Ingeniería de Sistemas

Acreditado de Alta Calidad

"Educación y Tecnología con Compromiso Social"

$$T(n) = 2 + k + 2 + 2 + n + 1 + \sum_{i=0}^{n-m} (1 + \sum_{i=0}^{m-1} (2 + k + 3 + k + 4 + k + 3 + 1) + 1) + 1 + 1$$

$$T(n) = 9 + n + k + \sum_{i=0}^{n-m} (2 + \sum_{i=0}^{m-1} (13 + 3k))$$

$$T(n) = 9 + n + k + \sum_{i=0}^{n-m} (2 + 13m + 3km)$$

$$T(n) = 3kmn + 13mn + 3m + 3n + k + 9$$

$$O(n) = nm$$

Line	Code	Operations Elementals
1	**public static int[] preproceso(String subcadena) {**	
2	int[] tablaDeSaltos = new int[256];	1 + 1 + k
3	for (int i = 0; i < 256; i++) {	Σ[desde i=0, hasta 256](
4	tablaDeSaltos[i] = subcadena.length();	1 + k
5	}	)
6	for (int i = 0; i < subcadena.length() - 1; i++) {	Σ[desde i=0, hasta n-1](
7	tablaDeSaltos[subcadena.charAt(i)] = subcadena.length() - 1 - i;	1 + k + 1
8	}	)
9	return tablaDeSaltos;	1
10	}	
11	**T(n)**	3n+kn+k+519
12	**O(n)**	**n**

$$T(n) = 1 + 1 + k + 1 \sum_{i=0}^{256} (1 + k + 1) + 1 + 1 + \sum_{i=0}^{n-1} (1 + k + 1 + 1) + 1 + 1$$

$$T(n) = 7 + k + 512 + 256k + 3n + kn$$

$$T(n) = 3n + kn + k + 519$$

$$O(n) = n$$

CONCLUSIÓN DE COSTE:

Caso	Complejidad
Promedio	O(n)
Peor	O(nm)

Para el caso de que la longitud de la subcadena "m" sea tan larga como el texto "n", se tendría un coste de O de n cuadrado, O(n^2).