UNIDAD PROFESIONAL INTERDICIPLINARIA DE INGENIERÍA CAMPUS ZACATECAS.

Práctica 2

Análisis y diseño de algoritmos

Autor:
Emilio de Jesús Ureño Padilla

Fecha:
21-Octubre-2023

1 Introducción

Para esta práctica se nos pide una implementación del método de ordenamiento Quicksort, se empleará para que mida su rendimiento a través de la medición de tiempos de ejecución en arreglos aleatorios y visualicen los resultados mediante gráficos, esto nos ayudará a comprender de mejor manera el funcionamiento del algoritmo.

2 Desarrollo

2.1 Implementación

El algoritmo Quicksort se utiliza en este caso a través de la función quicksort que se define en el código.

La función quicksort toma un arreglo (arr) como entrada y lo ordena de manera recursiva utilizando el algoritmo Quicksort.
Funciona de la siguiente manera dentro del código:

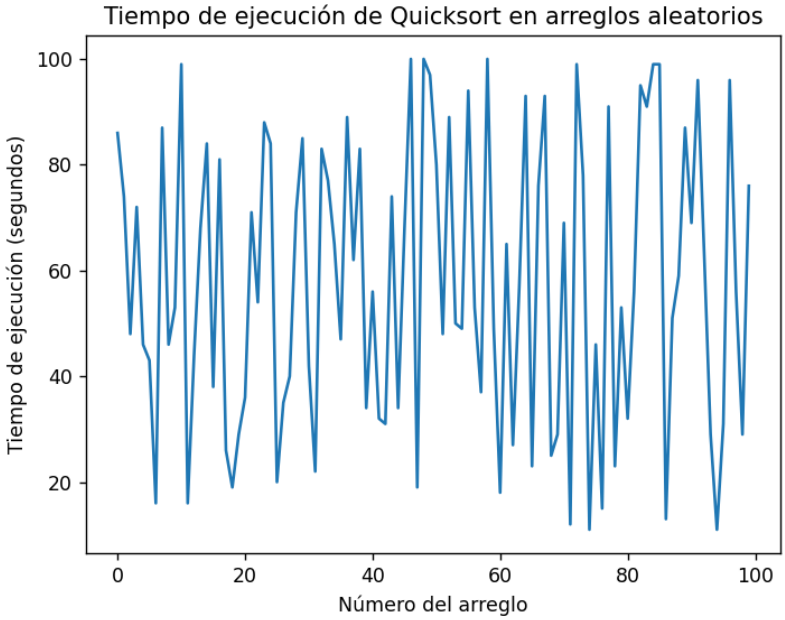
- Si el tamaño del arreglo (arr) es igual o menor que 1 (es decir, está vacío o tiene un solo elemento), se considera que ya está ordenado y se devuelve tal como está.
- De lo contrario, se elige un elemento del arreglo como pivote. En este caso, el primer elemento del arreglo (arr[0]) se toma como pivote.
- Se crean dos listas vacías: left y right. Estas listas se utilizarán para almacenar elementos menores que el pivote (left) y elementos mayores o iguales al pivote (right).
- Luego, se recorre el arreglo arr a partir del segundo elemento (índice 1) hasta el último elemento (índice len(arr) - 1).

  Si un elemento es menor que el pivote, se agrega a la lista left. Si es mayor o igual al pivote, se agrega a la lista right. Se llama recursivamente a la función quicksort en las listas left y right.
- Finalmente, se combinan los resultados de las llamadas recursivas de quicksort de la siguiente manera:

  Los elementos de left (ordenados) se concatenan con el pivote. Luego, se concatenan con los elementos de right (ordenados).

Esta combinación de pasos se repite de forma recursiva hasta que todos los subarreglos sean de tamaño 1 o 0, lo que garantiza que todo el arreglo esté ordenado.

2.2 Resultados



En la gráfica podemos observar los tiempos de los 100 arreglos, en el eje X tenemos el número del arreglo y en el eje y tenemos el tiempo que tarde en ordernarse.

2.2.1 Mejores 3 casos

Array 27: 0.0009965896606445312 segundos.
Array 27: [3, 4, 5, 7, 11, 15, 17, 20, 21, 26, 34, 35, 36, 36, 40, 41, 42, 43, 47, 48, 49, 49, 51, 58, 58, 59, 61, 63, 64, 81, 89, 90, 90, 94, 100]

Array 1: 0.0010008811950683594 segundos.
Array 1: [1, 3, 4, 4, 5, 6, 11, 12, 12, 12, 13, 13, 14, 15, 16, 16, 16, 18, 19, 20, 20, 21, 22, 22, 24, 25, 27, 30, 31, 31, 32, 33, 35, 35, 37, 37, 38, 38, 38, 42, 44, 45, 46, 47, 47, 48, 49, 51, 53, 56, 58, 59, 60, 60, 61, 61, 66, 68, 70, 71, 75, 76, 76, 77, 77, 78, 79, 92, 96, 96, 96, 98, 100, 100]

Array 4: 0.0010011196136474 61 segundos.
Array 4: [3, 4, 4, 5, 7, 8, 9, 9, 11, 12, 13, 14, 14, 17, 18, 19, 20, 20, 23, 23, 24, 26, 26, 27, 27, 28, 29, 34, 37, 37, 39, 41, 42, 43, 44, 44, 45, 46, 50, 51, 51, 53, 57, 57, 61, 62, 64, 65, 66, 66, 67, 69, 71, 74, 78, 80, 82, 85, 86, 88, 88, 90, 94, 94, 95, 95, 96, 97, 97, 97, 98, 99]

2.2.2 Peores 3 casos

Array 11: 0.001002192497253418 segundos.
Array 11: [1, 1, 1, 4, 5, 6, 6, 6, 8, 9, 9, 10, 11, 11, 11, 13, 13, 14, 17, 17, 18, 20, 21, 21, 24, 26, 27, 28, 29, 29, 30, 30, 31, 33, 33, 34, 35, 35, 36, 36, 37, 38, 39, 41, 43, 44, 46, 48, 49, 50, 51, 53, 53, 55, 56, 56, 56, 56, 58, 58, 58, 59, 60, 60, 60, 65, 66, 68, 68, 70, 72, 72, 76, 76, 77, 79, 80, 80, 82, 83, 85, 86, 86, 86, 86, 88, 88, 89, 89, 90, 92, 92, 94, 94, 94, 95, 97, 98, 98]

Array 9: 0.0010020732879638672 segundos.
Array 9: [1, 2, 2, 2, 3, 4, 4, 7, 7, 9, 11, 11, 14, 14, 14, 15, 16, 16, 16, 17, 17, 18, 19, 20, 20, 21, 26, 28, 29, 29, 31, 33, 33, 34, 34, 34, 36, 37, 40, 40, 40, 41, 43, 47, 48, 49, 54, 54, 56, 58, 58, 59, 59, 59, 59, 62, 63, 65, 67, 68, 68, 68, 71, 71, 73, 73, 74, 74, 76, 77, 77, 77, 78, 79, 79, 80, 80, 81, 89, 89, 89, 91, 92, 94, 95, 99, 100]

Array 20: 0.0010020732879638672 segundos.
Array 20: [3, 3, 4, 5, 5, 6, 6, 7, 8, 12, 12, 14, 14, 15, 15, 15, 16, 16, 18, 18, 20, 20, 22, 22, 23, 24, 25, 28, 28, 30, 31, 34, 34, 35, 38, 40, 40, 41, 41, 42, 42, 42, 45, 46, 48, 50, 51, 51, 52, 56, 58, 58, 59, 60, 61, 62, 63, 67, 71, 72, 73, 74, 74, 74, 76, 77, 77, 77, 77, 78, 79, 79, 82, 82, 82, 82, 84, 85, 87, 90, 91, 91, 92, 92, 95, 97, 98, 100]

3 Conclusión

Es un algoritmo que ayuda a reducir demasiado el tiempo de ejecución de muchas cosas, por ejemplo en este caso al imprimir arreglos de manera aleatoria con valores del 10 al 100, imprimir eso 100 veces con otros métodos de ordenamiento tarda casi el doble o el triple que con el algoritmo Quicksort; es eficiente y divide el arreglo en subarreglos más pequeños para ordenarlos por separado, lo que lo hace más rápido que algunos otros algoritmos de ordenamiento.

4 Referencias