Quick Sort

Erick Gonzalez Parada ID: 178145
Matemáticas discretas, Universidad de las Américas Puebla, Puebla, México

October 1, 2023

Abstract

Análisis de Quick Sort.

Keywords: pivote divide & conquer

1 Introducción

Quick Sort en c++

```
1  int partition(std::vector<int>& arr, int low, int high) {
2      int pivot = arr[(low + high) / 2];
3      int i = low - 1;
4      int j = high + 1;
5      while (true) {
6          do {
7              i++;
8          } while (arr[i] < pivot);
9          do {
10             j--;
11         } while (arr[j] > pivot);
12         if (i >= j)
13             return j;
14         std::swap(arr[i], arr[j]);
15     }
16 }
17
18 void quickSort(std::vector<int>& arr, int low, int high) {
19     if (low < high) {
20         int pi = partition(arr, low, high);
21         quickSort(arr, low, pi);
22         quickSort(arr, pi + 1, high);
23     }
24 }
25
26
27 int main() {
28     std::vector<int> arr = { 10, 7, 8, 9, 1, 5 };
29     int n = arr.size();
30     quickSort(arr, 0, n - 1);
31     std::cout << "Array ordenado: \n";
32     for (int i = 0; i < n; i++)
33         std::cout << arr[i] << " ";
34     return 0;
35 }
```

Figura 1: C++ Quicksort

Como podemos observar, el código del Quicksort, denotado por el script de la figura 1 funciona de la siguiente manera, primero divide una matriz grande en dos subarrays más pequeños y luego ordena recursivamentemente los subarrays. Básicamente, tres pasos están involucrados en todo el proceso, la selección del pivote, la división, y la repetición hasta que quedan ordenados.

Mejor caso

El mejor caso es cuando el pivote divide limpiamente en dos partes iguales, esto hará que tarde menos.

Peor caso

El peor caso es cuando el arreglo esta casi ordenado, haciendo que el proceso de comparación haga operaciones innecesarias.

2 Función de relación de recurrencia

Mejor caso

Como mencionado, el mejor caso es que el pivote parte el arreglo en dos subarrays de tamaño n/2 "perfectos" de tal manera que el procesamiento de ellos solo se lleva n tiempo, con la siguiente información nos genera la siguiente ecuación de recurrencia que resolveremos con el teorema maestro

$$T(n) = 2T(\frac{n}{2}) + n$$

Donde $a = 2$, $b = 2$, *f(n)* = n, $n^{\log_b(a)}$ = n. por lo que tenemos el caso dos del teorema maestro que dice If $f(n) = \theta(n^{\log_b(a)})$ then $T(n) = \theta(n \log_2(f(n))) = \theta(n \log(n))$.

Como es el mejor caso entonces tenemos que $\Omega(n \log(n))$ & sorprendentemente y como dato extra el caso promedio no cambia debido a que una ligera optimización con respecto a la selección del pivote puede frecuentar este caso.

Peor caso

$$T(n) = 2T(\frac{n}{2}) + n^2$$

donde $a = 2$, $b = 2$, *f(n)* = n^2, $n^{\log_b(a)}$ = n. por lo que ahora tenemos el caso 3 donde f(n) es mayor que $n^{\log_b(a)}$ entonces If $f(n) = \Omega(n^{\log_b(a)+\epsilon})$ then $T(n) = \Theta(f(n)) = \theta(n^2)$.

Como es el peor caso entones tenemos que $O(n^2)$.

3 Invariante de ciclo

El invariante del ciclo es que la variable *pi* contiene el valor del pivote correspondiente a esa iteración y eventualmente quedara ordenada.

inicialización

Antes de entrar el ciclo nosotros determinamos donde queremos que sea nuestro pivote e independientemente de la elección lo que sucederá en la siguiente fase es que como nuestro pivote es nuestro pivote pues se harán las evaluaciones con nuestro pivote de tal manera que quedara ordenado.

mantenimiento

Durante el ciclo como mencionamos es el proceso en donde nuestro pivote quedara ordenado, OJO la invariante de ciclo es que la variable *pi* siempre contenga el pivote, no se esta detallando que la invariante también es que esta ya este ordenada por lo que si nuestra invariante fuese falsa, no estaríamos ordenando el array.

finalización

Al finalizar se mantuvo verdadero siempre, la variante pi nunca tuvo un valor erróneo.

4 Conclusiones (problema real)

a pesar que de que te pueden preguntar su implementación en una entrevista, imagina que trabajas en una empresa de logística y tienes una gran cantidad de paquetes que deben ser entregados a diferentes destinos en todo el país. Los paquetes están mezclados y necesitas organizarlos en el orden correcto para optimizar las rutas de entrega y minimizar el tiempo de viaje.
En este ejemplo se puede ocupar sencillamente quicksort y bum listo.