Problema 1: Kaio Ken

Explicación:

Tenemos N elementos y el problema consiste en dividir esos elementos en 2 grupos, no necesariamente la mitad para un lado y la mitad para el otro (aunque en la resolución del problema veremos que esto nos favorece), con el siguiente criterio de división: repartir dichos elementos para que cada uno esté “enfrentado” a cada otro elemento al menos una vez. Al “enfrentarse” nos referimos a que si tenemos únicamente 2 elementos A y B entonces la manera de “enfrentarlos” va a ser repartiendo en un grupo el elemento A y en otro grupo el elemento B, aquí ampliamos con un ejemplo para N = 3:

Elementos:= ABC

Para resolver el problema podemos enfrentar a A con B y C, de esta manera nos queda únicamente enfrentar a B con C, entonces podemos dividir los 3 elementos en una segunda instancia pero esta vez con B por un lado, A y C por el otro. Cabe destacar que también podemos ubicar a C por un lado, y A y B por otro, y también cumpliríamos con la resolución, pero no sería minimal que es lo que pide el problema, en ese caso o bien optamos por poner a B solo o bien optamos por poner a C solo.

Complejidad: O(N Log2(N)) pues recorre todos los elementos del vector porque los tiene que repartir según corresponda, pero la serie de particiones sólo se dará una cantidad logarítmica de veces que es lo necesario para satisfacer las condiciones del problema.

Pseudocódigo:

Int rango := log2(n) + 2

If (n == 2(rango -2 )) rango := rango - 1

Para i := 1 hacer

Para x := 0 hacer

Si x mod 2i < 2(i – 1) entonces

el elemento va al Grupo 1

Else

El elemento va al grupo 2

Fin Si

X++

Fin Para

I++

Fin Para

Análisis experimental:

Para el análisis consideramos necesario experimentar únicamente aumentando el valor de entrada, ya que no tenemos ni mejor caso ni peor caso porque el algoritmo sin excepciones recorre todo el vector una cantidad logarítmica de veces según la entrada N, con lo cual presentamos un gráfico mostrando el valor de la entrada aumentando en potencias de 2. Con lo cual se deduce que, cuanto mayor es el tamaño de la entrada, más ciclos de CPU son necesarios y por tanto más tiempo.

Apéndice:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | for (i = 1; i < rango; i++) { | |  |  |  | | --- | |  | |  | |  | |  | | for (x = 0; x < n; x++) { | |  | |  | | if ((x % (int)pow(2, i)) < pow(2, i - 1)) { | |  | |  | | cout << GRUPO\_1; | |  | |  | | } | |  | |  | | else { | |  | |  | | cout << GRUPO\_2; | |  | |  | | } | |  | |  | | if (x < n - 1) cout << " "; | |  | |  | | } | |  | |  | | cout << endl; | |  |  |  | | --- | | } | |