Programación eficiente - Tarea 6 30 de Octubre del 2018
Miguel Angel Ruiz Ortiz

Divide y vencerás: Fuerza de N puntos.

Dados N puntos en $\mathbb{R}^2$, calculemos la fuerza de cada uno. La fuerza de un punto (x, y) es la cantidad de puntos distintos de (x, y) de los N dados que tienen abcisa y ordenada mayor o igual que x e y, respectivamente. Supongamos sin pérdida de generalidad que tenemos en un arreglo $P[1, \dots, N]$ los N puntos ordenados por la coordenada x. Denotemos $P[i].x$ y $P[i].y$ a la abcisa y ordenada del punto $P[i]$, respectivamente.

Supongamos $force(P[i])$ es la fuerza del punto $P[i]$. Al iniciar el algoritmo, $force(P[i]) = 0$ para todo i. Cuando se haga referencia a la fuerza de los puntos $P[i, \dots, j]$ nos referiremos a la fuerza de dichos puntos sin considerar los puntos $P[1, \dots, N] \setminus P[i, \dots, j]$ (permitamos el abuso de notación). La idea del algoritmo será calcular la fuerza de los puntos $P[1, \dots, N]$ a partir de la fuerza de los puntos $P[1, \dots, M]$ y de los puntos $P[M + 1, \dots, N]$, con $M = (N + 1)/2$.

Queremos calcular la fuerza de los puntos $P[i, \dots, j]$ y supongamos tenemos calculado la fuerza de los puntos $P[i, \dots, m]$ y $P[m + 1, \dots, j]$, con $m = (i + j)/2$. También supondremos que los arreglos $P[i, \dots, m]$ y $P[m + 1, \dots, j]$ están ahora ordenados por la coordenada y. Como al principio teníamos el arreglo ordenado por la coordenada x, entonces los puntos en el primer arreglo tienen abscisa menor o igual que los puntos en el segundo arreglo.

Notemos que la fuerza de los puntos $P[i, \dots, j]$ ya está casi calculada, de hecho, $force(P[k])$ para $m < k \leq j$ ya está calculada. Nos falta actualizar $force(P[k])$ para $i \leq k \leq m$. Sea $k \in \{i, \dots, m\}$ y sea $r \in \{m+1, \dots, j\}$ el menor tal que $P[k].y \leq P[r].y$. Notemos entonces $P[k]$ domina a los puntos $P[r], P[r+1], \dots, P[j]$, y entonces debemos actualizar la fuerza de $P[k]$ como $force(P[k]) += j - r + 1$.

Para realizar lo anterior y ordenar al mismo tiempo $P[i, \dots, j]$ con respecto a la coordenada y de los puntos, podemos realizar lo siguiente. Como en el MergeSort, en un arreglo auxiliar $aux[\,]$ iremos poniendo los puntos de $A[\,] := P[i, \dots, m]$ y $B[\,] := P[m + 1, \dots, j]$ de manera en que $aux[\,]$ tenga esos puntos ordenados por la coordenada y. Al principio, debemos escoger si el primer elemento de A va primero que el elmento de B o viceversa. Si estamos en el primer caso, agregamos $A[1]$ a $aux[\,]$ y notemos que entonces debemos actualizar $force(A[1]) += j - m$. Luego redefinimos $A[] = P[i + 1, \dots, m]$. Si estamos en el segundo caso, sólo debemos agregar $B[1]$ a $aux[\,]$ y redefinir $B[\,] = P[m + 2, \dots, j]$. Y así sucesivamente iremos rellenando $aux[\,]$ y a su vez calculando la fuerza de los puntos $P[i, \dots, m]$.

El caso base de la recursión anterior es cuando $i = j$, en donde no tenemos que hacer nada pues el arreglo $P[i, \dots, j] = P[i]$ ya está ordenado y la fuerza de $P[i]$, que es 0, ya también estaba calculada. Tenemos entonces un algoritmo recursivo para calcular la fuerza de los puntos $P[1, \dots, N]$.

$\square$