CC4102 - Control 2

Prof. Gonzalo Navarro

9 de Noviembre de 2020

P1 (2.0 pt)

Se quiere una estructura de datos que implemente un arreglo permitiendo las siguientes operaciones:

1. Insertar un elemento al final del arreglo.
2. Insertar un elemento al comienzo del arreglo.
3. Acceder al k-ésimo elemento del arreglo.

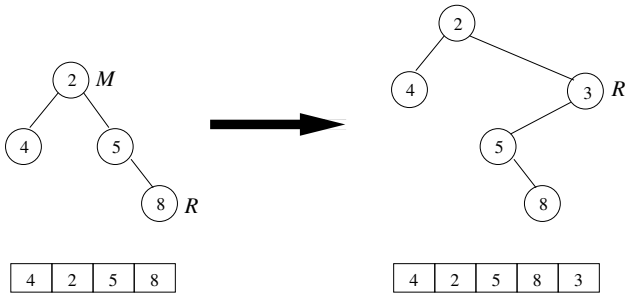
Si el arreglo contiene n elementos, debe ocupar espacio $O(n)$.

Diseñe y analice una solución que resuelva las dos primeras operaciones en tiempo amortizado constante y el acceso en tiempo constante en el peor caso.

P2 (3.0 pt)

Dado un arreglo de enteros distintos $A[1, n]$, un *árbol cartesiano* es un árbol binario de n nodos. Si el mínimo de A está en $A[i]$, entonces la raíz del árbol cartesiano corresponde a $A[i]$, el hijo izquierdo al árbol cartesiano de $A[1, i-1]$ y el hijo derecho al árbol cartesiano de $A[i+1, n]$. Cuando el rango de A se hace vacío el árbol cartesiano es vacío.

Considere el siguiente algoritmo para construir el árbol cartesiano. Se procesa $A[1, n]$ de izquierda a derecha, manteniendo un puntero R al nodo de más a la derecha del árbol. Para cada nuevo $A[i]$, se revisa desde R hacia arriba hasta encontrar el primer nodo M con un elemento menor a $A[i]$. Se inserta $A[i]$ como hijo derecho de M, y la rama recorrida como su hijo izquierdo. Luego el nodo de $A[i]$ pasa a ser R.



Use análisis amortizado para mostrar que este algoritmo es $O(n)$.

P3 (1.0 pt)

Suponga que tiene una situación en la que necesita colas de prioridad. Para cada uno de los siguientes casos, debe indicar si convendría usar un heap común, una cola binomial, una de Fibonacci o si da lo mismo cuál usar. Argumente brevemente en cada caso.

1. Necesita unir colas frecuentemente. La estructura se utilizará en una aplicación para corregir los movimientos de un vehículo autónomo en tiempo real.

2. No necesita unir colas frecuentemente. La aplicación inserta una gran cantidad de objetos y se pretende ir obteniendo unos pocos más relevantes en medio de las inserciones.
3. Usará la estructura para insertar todos los elementos de un conjunto y luego obtenerlos ordenados al extraer repetidamente el mínimo.