



ALGORITMOS Y COMPLEJIDAD

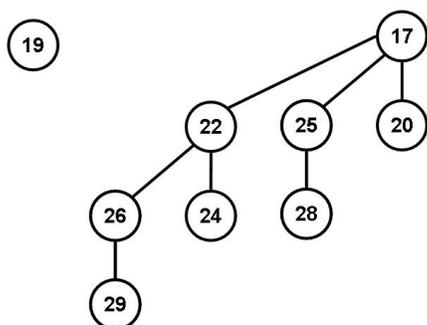
TRABAJO PRÁCTICO 9

Análisis Amortizado

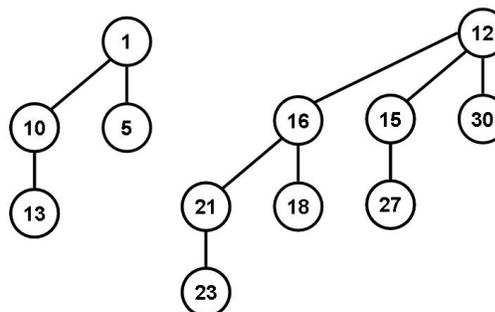
Primer cuatrimestre de 2017

1. Colas Binomiales

- Definir Cola Binomial e indicar cómo implementar en forma eficiente dicha estructura.
- Dar un algoritmo para la operación $Minimo(C)$ la cual retorna el mínimo elemento de la cola binomial C .
- Dar un algoritmo para la operación $DisminuirClave(C, x, k)$ la cual disminuye el valor de x reemplazándolo por el nuevo valor k .
- Dar un algoritmo para la operación $Unir(C_1, C_2)$ la cual une (mezcla) dos colas binomiales. Indicar claramente todos los casos que deben considerarse al momento de proceder con la mezcla.
- ¿Cuál es el resultado de realizar la mezcla/unión de las colas binomiales Cb_1 y Cb_2 ?



(a) Cola Binomial Cb_1



(b) Cola Binomial Cb_2

- Indicar cómo se pueden realizar las siguientes operaciones, utilizando las operaciones definidas anteriormente.
 - $Insertar(C, x)$: inserta un nuevo nodo x en la cola C
 - $EliminarMin(C)$: elimina el mínimo elemento de C , y
 - $Eliminar(C, x)$: elimina el nodo x de C ,

2. Skew Heaps (Heaps asimétricos)

- ¿Qué características tienen los *Skew heaps*?
- Dar un algoritmo para la operación $Mezclar(S_1, S_2)$ que mezcla dos *Skew heaps*.
- ¿A qué se denominan *nodos pesados* y *nodos livianos*? ¿Cómo varía la cantidad de nodos pesados luego de la ejecución de la operación $Mezclar$?

3. ¿Qué se entiende por análisis amortizado? Describa brevemente el método del potencial.

4. ¿Qué podemos afirmar si en un análisis amortizado luego de N operaciones el valor del potencial $\Phi(D_N)$ es menor que el potencial inicial $\Phi(D_0)$?

5. Considere el problema de implementar una cola con dos pilas comunes.

- a) Dar los algoritmos para las operaciones: `ponerEnCola(e)` y `quitarDeCola`. Analizar su tiempo de ejecución.
- b) Dar una función de potencial adecuada y analizar el costo amortizado de cada operación.

6. *Tablas Dinámicas.*

Sea $\alpha(T_i)$ la *tasa de ocupación* de una tabla dinámica en el momento i , definida como $\frac{\text{elementos}(T_i)}{\text{capacidad}(T_i)}$. Observar que entonces la expansión ocurre cuando $\alpha(T_i) = 1$.

- a) Mostrar que si se introduce la operación borrado en donde la tabla se contrae si $\alpha(T_i) < 0,5$, la función potencial vista en teoría no sirve para demostrar un costo amortizado de $O(1)$ para ambas operaciones.
- b) Considerar una situación idéntica a la anterior, pero en donde la tabla se contrae solamente cuando $\alpha(T_i) < 0,25$. Demostrar que la función potencial Φ que se define a continuación permite demostrar un costo amortizado de $O(1)$ para ambas operaciones:

$$\Phi(T) = \begin{cases} 2 \times \text{elementos}(T) - \text{capacidad}(T), & \text{si } \alpha(T) \geq 0,5 \\ \text{capacidad}(T)/2 - \text{elementos}(T), & \text{si } \alpha(T) < 0,5 \end{cases}$$

7. Analizar el tiempo de ejecución de las operaciones definidas para las estructuras de datos *Fibonacci Heaps* y *Skew Heaps*.

8. *Fibonacci Heaps*

- a) Definir Fibonacci Heap e indicar cómo implementar en forma eficiente dicha estructura. Describir la política del marcado de los nodos y cortes en cascada. ¿Qué sucedería si no se sigue esta política?
- b) Describir cómo se realizan las siguientes operaciones.
 - *Insertar*(F, x): inserta un nuevo nodo x en el fibonacci heap F
 - *Unir*(F_1, F_2): crea y retorna un nuevo fibonacci heap que contiene los elementos de los heaps F_1 y F_2 (estos son destruidos como resultado de la operación)
 - *Eliminar*(F, x): elimina el nodo x del heap F .
- c) Dar los algoritmos correspondientes para cada una de las siguientes operaciones.
 - *EliminarMinimo*(F): elimina el elemento de F que tiene la clave con menor valor,
 - *DisminuirClave*(F, x, k): disminuye el valor de x reemplazándolo por el nuevo valor k .
- d) Analizar cómo varía la cantidad de árboles y la cantidad de nodos marcados luego de la ejecución de cada una de las operaciones que soportan los Fibonacci Heaps.
- e) Determinar paso a paso el Fibonacci Heap que se obtiene como consecuencia de la siguiente secuencia de operaciones:
 - 1) Comenzando con un heap vacío, insertar en secuencia los elementos 10, 13, 18
 - 2) Eliminar el mínimo elemento
 - 3) Insertar en secuencia los elementos 14, 16, 20
 - 4) Eliminar el mínimo elemento
 - 5) Insertar en secuencia los elementos 5, 6, 3, 8, 9
 - 6) Eliminar el mínimo elemento
 - 7) Disminuir la clave del nodo 9 al valor 2
 - 8) Unir el heap hasta aquí obtenido con F_1
 - 9) Disminuir la clave del nodo 45 al valor 33
 - 10) Disminuir la clave del nodo 40 al valor 32
 - 11) Eliminar el mínimo elemento

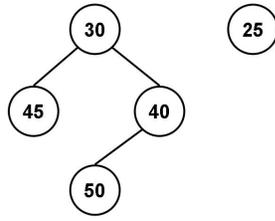


Figura 1: Fibonacci Heap F_1

Referencias

- [BB96] Gilles Brassard and Paul Bratley. *Fundamentals of Algorithmics*. Prentice Hall, 1996.
- [CLRS09] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction To Algorithms*. The MIT Press, 3rd edition, 2009.