Universidad del Valle de Guatemala
Análisis y Diseño de Algoritmos - Sección 10
María Fernanda Estrada 14198
13/05/2020

Tarea 5
Análisis amortizado

1. **Agregamos al contador binario una operación de decremento: *Decrement*. Identifique el worst-case scenario realista para una secuencia de n operaciones *Increment* y *Decrement* combinadas, y provea una cota superior para su tiempo de ejecución. Suponga que el contador trabaja con una lista de k bits.**
   El worst-case scenario sería aplicar las n operaciones a cada bit; es decir, que se modifiquen los k bits con n operaciones intercaladas de incremento y decremento. Tomando esto en cuenta, la cota superior del tiempo de ejecución sería O(nk).

2. **Supongamos que a una pila con multipop agregamos la operación multipush(k, A) que hace una secuencia de operaciones push con los primeros k elementos en el arreglo A. En este caso, ¿se mantiene el costo amortizado de $O(1)$ por operación? Explique.**
   No se mantiene porque si hacemos una secuencia de multipush y multipop intercaladas, el costo de esta secuencia sería de $O(n^2)$. Entonces, al dividir por n para encontrar el costo amortizado por operación, se tiene que es de $O(n^2)/n = O(n)$.

3. **Considere una pila común y corriente cuyo tamaño nunca excede k. Luego de k operaciones push y/o pop se efectúa un backup automático, copiando toda la pila. El costo de copiar m elementos es m. Demuestre con el accounting method que cualquier secuencia de n operaciones entre push y pop (con backups cada k operaciones) costará $O(n)$.**
   Primero, asignamos un costo amortizado de 2 a la operación push para asegurarnos que cuando suceda un backup tendremos de más para pagar por la copia de cada elemento. Luego, asignamos a pop un costo amortizado de 1, ya que como es su costo real no afectará el extra que agregó la operación push. Entonces, en cualquier secuencia de k operaciones, se tendrán k créditos para compensar la operación de backup. Una secuencia de n operaciones costará si mucho 2n, produciendo una cota superior de O(n). Esto es si todas las operaciones son push.

4. **Suponga que al contador binario de los ejemplos se le agrega la operación reset que busca y convierte todos los 1 en 0, uno por uno a partir del bit menos significativo. Demuestre con el accounting method que cualquier secuencia de n operaciones entre increment y reset toma un tiempo de ejecución de $O(n)$. Considere que el contador inicia desde 0 y que cada revisión y cada modificación de un bit toma $\Theta(1)$.**

   Nos interesa saber qué costo tendrá increment, aparte del 1 inicial asignado. Lo primero que consideramos es que el costo de increment deberá ascender a 2 porque debe incluir un crédito para pagar el 0 que regresa a 1 cuando se aplique. De esta forma garantizamos que para cada 1 habrá un crédito almacenado que pague por regresar a 0. Luego, nuevamente el costo de increment deberá ascender a 3 para acumular dos créditos por cada bit que se convierte en 1, con el propósito de pagar por una operación de reset. Esto nos garantiza que por cada bit modificado al menos una vez desde el último reset tendremos un crédito para pagar por un reset. Por último, se hace un nuevo incremento al costo para llegar a 4. Esto es porque no es conveniente que reset recorra todo el arreglo porque no podemos asegurar que se tengan k créditos acumulados. Debemos llevar un registro de cuál es el bit más significativo y verificar si la última posición modificada es más grande que la anterior; si sí es así, se actualiza la posición con costo 1.

   Ahora, con la operación reset tendrá costo 0 por el bit más significativo que actualizará cuando se ejecute. Por consiguiente, cualquier secuencia de n operaciones tendrá un costo amortizado de 4n => O(n).

5. **Tenemos una función de potencial Φ tal que $\Phi(D_i) \geq \Phi(D_0)$ para todo i, pero $\Phi(D_0) \neq 0$. Defina una función Φ' tal que $\Phi'(D_0) = 0$, $\Phi'(D_i) \geq \Phi'(D_0)$, $\forall_i \geq 1$, y que los costos amortizados obtenidos con Φ' sean los mismos que con Φ.**

   Si queremos que los costos amortizados se mantengan iguales, comenzamos a plantear lo siguiente

$$\Phi(D_j) - \Phi(D_i) = \Phi'(D_j) - \Phi'(D_i)$$

$$\Phi'(D_j) - \Phi'(D_i) = \Phi(D_j) - \Phi(D_i) - \Phi(D_0) + \Phi(D_0) = (\Phi(D_j) - \Phi(D_0)) - (\Phi(D_i) - \Phi(D_0))$$

   Entonces, si $\Phi'(D_i) = \Phi(D_i) - \Phi(D_0)$, se tiene finalmente que

$$\Phi'(D_0) = \Phi(D_0) - \Phi(D_0) = 0, \Phi'(D_i) \geq \Phi'(D_0)$$

6. **Un min-heap binario es un árbol binario completo (todos sus niveles están llenos y sus hojas se llenan de izquierda a derecha) en donde cada nodo es menor que todos sus hijos. El min-heap tiene una operación de inserción llamada insert, pero consideremos además la función extract-min, que reemplaza la raíz por el último nodo del árbol y luego la intercambia con el menor de sus hijos. Supongamos que ambas operaciones tienen un tiempo de ejecución real $O(\log_2 n)$. Provea una función de potencial según la cual el costo amortizado de insert sea $O(\log_2 n)$ y el de extract-min sea $O(1)$. No olvide que el potencial siempre debe ser positivo (no necesariamente el cambio de potencial), y que el potencial inicial es idealmente cero. Demuestre que su función de potencial funciona.**

   Se toma como base el tiempo de ejecución real de ambas operaciones, el cual es $O(\log_2 n)$. Para una inserción, se necesita que la fórmula de costo amortizado sea $O(\log_2 n)$ y que sea $O(1)$ para el extract-min, por lo que se tienen las siguientes ecuaciones:

   $$a_i = O(log_2 n) + (\Phi(D_i) - \Phi(D_{i-1})) = O(log_2 n) \quad \text{#inserción}$$
   $$a_i = O(log_2 n) + (\Phi(D_i) - \Phi(D_{i-1})) = O(1) \quad \text{#extract-min}$$

   La primera expresión indica que el cambio de potencial debe ser $\log_2 n$ o menor. La segunda expresión nos requiere que el cambio de potencial sea negativo y de orden $\log_2 n$. En conclusión, necesitamos que nuestro cambio de potencial sea $-O(\log_2 n)$.

   Si consideramos la profundidad del i-ésimo nodo y la sumatoria de todos estos nivele, obtenemos que:

   $$\Phi(D_i) = \Sigma_{i=1}^{n} prof_i = \Sigma_{i=1}^{n-1} prof_i + O(log_2 n)$$

   Entonces, obtenemos que la función de potencial es:

   $$\Phi(D_i) - \Phi(D_{i-1}) = \Sigma_{i=1}^{n-1} prof_i - (\Sigma_{i=1}^{n-1} prof_i + O(log_2 n)) = - O(log_2 n)$$

   Para probar que funciona, reemplazamos en las primeras ecuaciones antes mencionadas y deben resultar en lo que se indica en el lado derecho.

   $$a_i = O(log_2 n) + (\Sigma_{i=1}^{n} prof_i + O(log_2 n) - \Sigma_{i=1}^{n} prof_i) = O(log_2 n) \quad \text{#inserción}$$
   $$a_i = O(log_2 n) - O(log_2 n) = O(1) \quad \text{#extract-min}$$