Auxiliar 5 - Más Análisis Amortizado y Algo de Dominios Discretos

CC4102 - Diseño y Análisis de Algoritmos
Profesor: Pablo Barceló Auxiliar: Jorge Bahamonde

24 de Abril del 2015

1. Suponga que se le entrega una implementación de un *stack*, en la que las operaciones PUSH y POP toman tiempo constante. Utilice estas estructuras para implementar una lista (*queue*) en la que las operaciones ENQUEUE y DEQUEUE tienen costo amortizado constante.

2. Un *quack* es una mezcla entre queues y stacks. Puede ser visto como una lista de elementos, escrita de derecha a izquierda, que soporta las siguientes operaciones:

   - QUACKPUSH(x) agrega x en el extremo izquierdo.
   - QUACKPOP(x) remueve y retorna el elemento más a la izquierda.
   - QUACKPULL(x) remueve y retorna el elemento más a la derecha.

   Implemente un quack usando tres stacks (idénticos a los del problema anterior) de modo que cada operación tenga un costo amortizado constante. Puede utilizar $O(1)$ memoria. Almacene los elementos sólo una vez en cualquiera de los 3 stacks.

3. Un *multistack* consiste en una serie (potencialmente infinita) de stacks $S_0, ..., S_{t-1}$ donde el j-ésimo stack puede almacenar hasta 3^j elementos. Todas las operaciones de *push* se realizan inicialmente sobre S_0.Cuando se desea *pushear* un elemento en un stack lleno S_j, se vacía este stack en el siguiente, S_{j+1} (posiblemente repitiéndose la operación de forma recursiva). Considere que las operaciones de *push* y *pop* en los stacks individuales tiene costo 1.

   - En el peor caso, ¿cuánto cuesta una operación de *push* en esta estructura?
   - Demuestre que el costo amortizado de una secuencia de n operaciones de *push* en un multistack inicialmente vacío es de $O(n \log n)$. Utilice la siguiente función de potencial:

     $$\Phi = 2 \sum_{j=0}^{t-1} N_j \cdot (\log_3 n - j)$$

     donde N_j es el número de elementos en el j-ésimo stack.
   - Demuestre lo mismo para cualquier secuencia de *pushes* y *pops*.

4. Se desea ordenar n puntos que pertenecen al círculo unitario según su distancia al origen, de menor a mayor. Diseñe un algoritmo que en promedio tome tiempo $O(n)$, suponiendo que los puntos se distribuyen de manera uniforme en este espacio.