Montículo de Fibonacci

César Ureña Toledano

9 de enero de 2022

Resumen

Algunos casos de prueba de la implementación de los montículos de Fibonacci y comparaciónes de su coste teórico con el dado por esta.

1. Introducción

La implementación que se ha usado para la implementación de los montículos es en base al pseudocódigo de las transparencias vistas en clase. Se ha realizado en Java y se ha usado el entorno de programación Eclipse. Se han implementado las operaciones de unión, mínimo, inserción, borrado del mínimo, y decrecer una clave suponiendo que se tiene un puntero al nodo de dicha clave.

Todos los archivos necesarios para la ejecución de esta implementación han sido provistos en la entrega.

2. Implementación y Pruebas

Todas las operaciones implementadas han sido probadas con casos de prueba para comprobar su correcto funcionamiento. Todas estas operaciones estan implementadas en la clase 'MonticuloFib' en el archivo 'MonticuloFib.java'. Para ver el estado del montículo después de cada operación, se ha implementado una operación 'display'. La obtención del mínimo no se ha inlcuido como prueba ya que es solo un acceso a un atributo de la clase y ya se sabe que es coste constante.

2.1. Inserción

Para empezar, se probó a insertar 5 elementos seguidos, unos más grandes que otros. Se inserta a la izquierda del mínimo (Ver Figura 1). Como vemos, siempre se inserta a la izquierda del mínimo. El 5 inicial es el mínimo y el dos se inserta a la izquierda que pasa a ser el mínimo por eso se representa el primero (La función 'display' empieza la representación de manera recursiva desde el elemento mínimo y cada nuevo parentesis al lado de un elemento significa un nuevo nivel y cada nivel se cierra con una flecha seguida de un paréntesis). Después se van insertando el resto de elementos que son mayores que el mínimo.

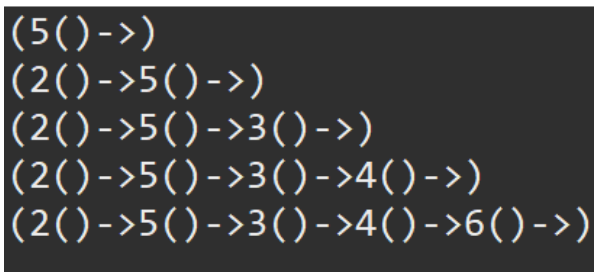

Figura 1: Ejemplo de inserciones consecutivas

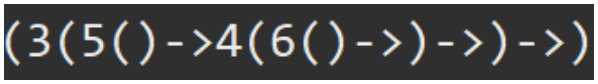

Figura 2: Ejemplo Borrado del Mínimo

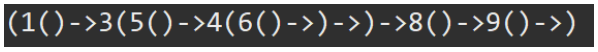

Figura 3: Otro ejemplo de inserción

2.2. Borrar el mínimo

La siguiente prueba es borrar el mínimo del montículo anterior, que es el elemento '2'. En los montículos de Fibonacci al eliminar el mínimo es cuando este se reequilibra, dejando una sola rama de cada grado (el grado esta representado por la cantidad de niveles que tenga). En este caso al borrar el mínimo, se quedan cuatro elementos, dos de ellos se van a juntar por tener grado 0 y después los otros dos siguientes. Al hacer esto, se forman dos de grado uno que se tienen que juntar en uno solo de grado 2, como vemos en la Figura 2

2.3. Inserción

Si repetimos una inserción vemos que se cumplen las condiciones dadas por el montículo. En este caso se insertan tres elementos y uno de ellos siendo menor que el actual mínimo con lo que pasa a ser el nuevo mínimo (Ver Figura 3).

2.4. Borrar el mínimo

Si repetimos un borrado del mínimo con el estado actual del montículo, se borraría el elemento '1' y se juntarían los dos elementos insertados anteriormente, que tienen ambos grado 0 y pasan a ser uno de grado 1 con el que ya se habia equilibrado en el borrado del mínimo anterior, de grado 2.

2.5. Secuencia de operaciones

En la siguiente Figura 5, se muestra la siguiente secuencia de operaciones: Insertar (10,12,30,7), Eliminar mínimo, Insertar (14), Eliminar mínimo, Eliminar mínimo. Como podemos ver, la secuencia de operaciones se hace de manera correcta con respecto a las operaciones de un montículo de Fibonacci.

2.6. Decrementar elemento

En el montículo resultante de las operaciones anteriores, se aplican dos operaciones de decrementar elemento. En la primera de ellas se decrementa el elemento '9' al '2' dejando al nodo '8' marcado ya que ha perdido uno de sus hijos. En la función 'display' se representa con dos asteriscos antes y después del elemento. Ej: *8*

En el montículo resultante de la primera operación de decrementar elemento, se vuelve a realizar un decremento del '10'(hijo del nodo marcado) al '4'. Al ser hijo del nodo marcado, se realiza un corte en cascada hasta que no haya nodos marcados o se llegue a la raiz (Ver Figura 6). Al cortarlo, se inserta a la izquierda del mínimo.

2.7. Unión de dos montículos

Para la siguiente pureba se crean dos montículos diferentes y se hace la unión de ambos. Esto es sencillo ya que es solo concatenar ambas listas encadenadas.

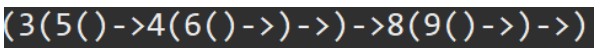

Figura 4: Otro borrado del mínimo

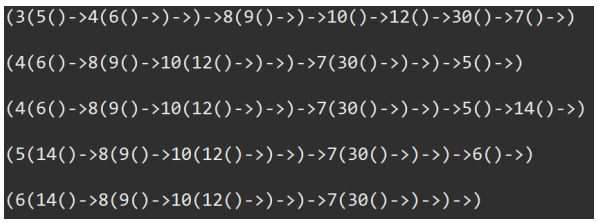

Figura 5: Secuencia de operaciones

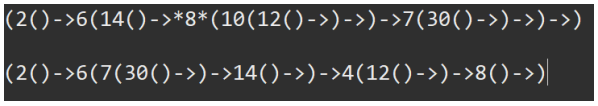

Figura 6: Dos operaciones decrementar elemento

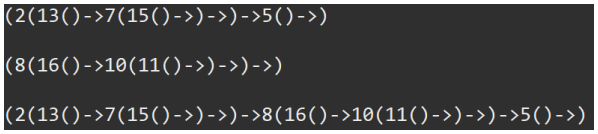

Figura 7: Unión de dos montículos

3. Complejidad

Usando gnuplot, se ha medido el coste real de las operaciones. Se ha probado a hacer la ejecucion desde 100 hasta 200000 elementos. Para todas las operaciones se ha repetido la ejecución bastantes veces y después se ha hecho una media del tiempo empleado en ellas. El archivo utilizado para realizar estas mediciones se incluye en la entrega.

3.1. Coste constante

Las operaciones unión, inserción, decrementar elemento y obtención del mínimo tienen coste teórico constante (El coste de la obtención del mínimo no se ha incluido ya que solo es un acceso a un atributo de la clase que ya sabemos que tiene conste constante).

En la grafica de la Figura 8 vemos que la unica que no tiene coste constante es la azul (1:4 en el archivo generado) que pertenece a la eliminación del mínimo. Unión, inserción y decrementar elemento estan en coste constante, la linea de conste constante en la gráfica donde están las tres superpuestas en practicamente coste 0. Con lo cual, vemos que el coste teorico constante se cumple.

3.2. Coste Logaritmico

Podemos ver que el coste de eliminar el mínimo crece lentamente con el número de elementos. En el peor de los casos el coste esta en O(n) pero vemos que el coste amortizado esta en O(log n) siendo n el número de elementos. Con lo cual, el coste teórico logaritmico en el número de elementos se cumple.

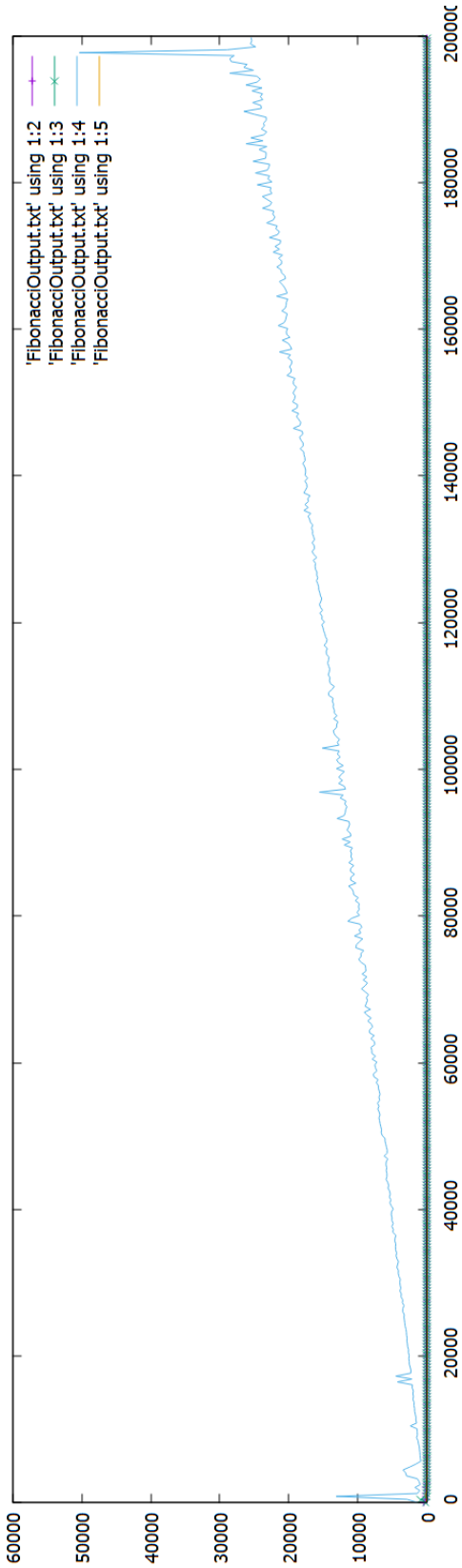

Figura 8: Grafica de los costes reales de la implementación