**Alberto Sánchez Gómez**

**Métodos Algorítmicos para la Resolución de Problemas**

**Montículo de Fibonacci**

El montículo de Fibonacci es una estructura de datos que tiene unos muy buenos tiempos para las operaciones de unión, inserción, decrecer una clave y obtener el mínimo. Todas estas tienen un coste amortizado constate. También tiene otras operaciones como la de borrar el mínimo o borrar un elemento arbitrario con un buen coste, pero este ya es logaritmo en el número de nodos en el montículo. A continuación, vamos a ver la implementación que he hecho y los costes que he obtenido.

Para la **implementación**, cada función la he implementado según indicaba el pseudocódigo presente en las transparencias, adaptándolo y convirtiéndolo a código. En mi caso, he usado el lenguaje C++.

Para la **medición de los tiempos**, he usado la librería “Windows.h”, y con las funciones que están en el .cpp de StartCounter() y getCounter() he medido los tiempos. Por lo que pude informarme, con esta librería y estas funciones obteníamos unos tiempos de medición lo más precisos posibles. He intentado obtener las gráficas del tiempo amortizado y del tiempo en el caso peor.

Las gráficas las he obtenido con *gnuPlot*. He creado unas tablas manualmente en unos ficheros .*dat*, que están en la carpeta subida al campus, donde he puesto los tiempos de cada operación según el tamaño del montículo. El eje de las X es el tamaño del montículo y el eje de las Y el tiempo que ha tardado.

Para el tiempo **amortizado** he ejecutado cada función 10.000 consecutivamente para ver cuánto tardaba esa operación en una secuencia de operaciones. En la tabla está el tiempo total de las 10.000 ejecuciones, para obtener el tiempo de una ejecución habría que dividir por 10.000.

Está gráfica se obtiene a partir del fichero “amortized.dat” y hay que ejecutar el siguiente comando en gnuPlot:

plot "amortized.dat" using 1:2 title 'Extract min' with lines,\

"amortized.dat" using 1:3 title 'Min' with lines,\

"amortized.dat" using 1:4 title 'Decrease key' with lines,\

"amortized.dat" using 1:5 title 'Union' with lines

Para generar la gráfica de una sola operación, habría que quitar “,\” del final de la línea y que al comienzo aparezca “plot”. Algo así:

plot "amortized.dat" using 1:4 title 'Decrease key' with lines

Gráfico, Gráfico de líneas

Descripción generada automáticamente

Cada punto de la gráfica lo he medido 3 veces y he hecho la media.

Vemos que las operaciones de unión, min y decrease key no se distinguen prácticamente. Todas ellas son de tiempo constante y cada una tiene un tiempo distinto, pero como el tiempo de extract min es mucho más alto, la escala hace que se vean juntas. El extract min debería ser logaritmo, y en mi gráfica es algo parecido, aunque no llega a ser del todo logarítmico. Incluso, había veces en los que me daban tiempos menores casos más grande que otros que eran más pequeños. Esto creo que puede ser porque cada vez que medía el tiempo de ejecución, daba un tiempo distinto, a veces con una diferencia de hasta 20 milisegundos. Esto no se si se debe a problemas de precisión del reloj de medición o a algo que estoy haciendo mal.

Para las operaciones constantes, he comprobado que daba igual el número de nodos que tuviese el montículo, por lo que he hecho una media con varios casos y le he puesto el mismo tiempo a todos los tamaños.

Para el tiempo en el **caso peor,** la gráfica correspondiente se ha realizado mediante el fichero “worst\_case.dat”. De la misma forma que la anterior, es una tabla con los mismos tamaños que antes, pero ahora el tiempo en el caso peor. La escala de los tiempos está en microsegundos, y el tiempo corresponde a una única ejecución de la función.

Los tiempos están medidos cuando la estructura está en el caso peor para cada operación. Para la operación de borrar el mínimo, se ha realizado cuando el montículo está con todos los nodos en la rootlist. Para el caso de obtener el mínimo y la unión son constantes también, por lo que tienen los mismos tiempos que en la anterior gráfica. Para el caso de decrecer clave, he probado con varios tamaños, pero el tiempo que obtengo siempre es muy variable ya que la ejecución de esa función tarda muy poco. He intentado crear la gráfica, pero debido al error de precisión y puede que otros factores, la gráfica puede que no sea muy fiable. El caso peor de esta función lo he probado con un puntero al último nodo insertado e inicializando todos los nodos con el atributo “mark” a true.

De igual manera que antes, se utiliza el mismo comando, pero cambiando el nombre del fichero a “worst\_case.dat”.

plot "worst\_case.dat" using 1:2 title 'Extract min' with lines,\

"worst\_case.dat" using 1:3 title 'Min' with lines,\

"worst\_case.dat" using 1:4 title 'Decrease key' with lines,\

"worst\_case.dat" using 1:5 title 'Union' with lines

Gráfico, Gráfico de líneas

Descripción generada automáticamente

El tiempo en el decrease key es muy inferior al de extraer el mínimo, por lo que la gráfica no se llega a observar.

Gráfico, Gráfico de líneas

Descripción generada automáticamente

Vemos que el borrado del mínimo da una gráfica lineal. No tiene una pendiente constante, pero sigue un patrón lineal. La gráfica del decrecer clave por lo que he comentado arriba no es muy fiable, pero podemos observar un patrón logarítmico más o menos, que era lo esperado.

En cuanto al formato de los **ficheros,** tienen la siguiente estructura:

* En la primera línea un número que indica si tenemos uno o dos montículos de entrada. Si ponemos dos, va a hacer la unión de ambos.
* Luego el número de nodos que vamos a insertar. Si son dos montículos, cada uno en una línea.
* Luego una secuencia de operaciones a realizar. Podemos poner todas las que queramos, siguiendo:
  + Imagen que contiene Interfaz de usuario gráfica

    Descripción generada automáticamente‘e’ para extraer el mínimo
  + ‘m’ para mostrar el mínimo por pantalla
  + ‘d’ junto a un número, para decrecer una clave a la que introducimos. Está función decrece el último nodo introducido en el montículo.

Se adjuntan varios ficheros cada uno con un caso de prueba. Los ficheros están numerados del 1 al 5, y están ordenados ascendentemente en complejidad.