Universidad Nacional San Agustin
Facultad de Ingeniería de Producción y Servicios
Departamento Académico de Ing. de Sistemas e Informática

UNSA
UNIVERSIDAD NACIONAL DE SAN AGUSTÍN DE AREQUIPA

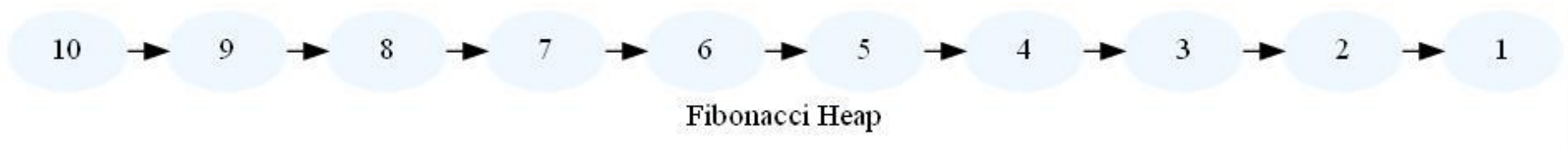
Fibonacci Heap

Programación y Análisis

Autor: Juan Manuel Soto Begazo
Josue Gabriel Sumare Uscca
Julio Enrique Yauri Ituccayasi
Docente: Cristian Jose Lopez Del Alamo
Estructura de Datos y Algoritmos
Fecha de entrega: 3 de diciembre de 2021
Arequipa, Peru

Insertamos 10 Nodos a nuestro Fibonacci Heap.

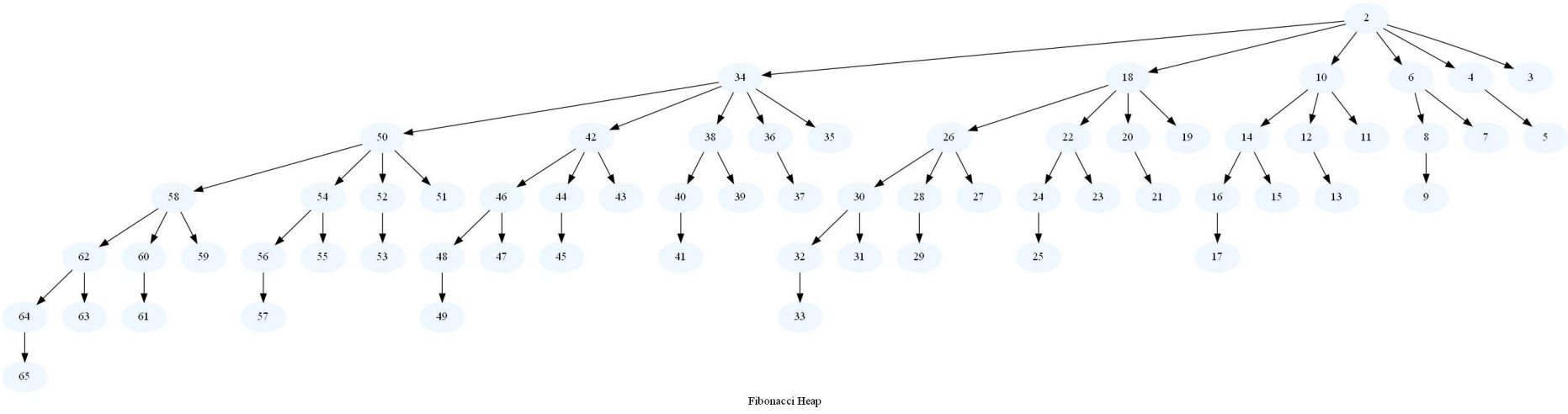
Como no realizamos ninguna operación adicional, la estructura que toma este Fibonacci Heap es la de una Lista Enlazada como podemos ver.



Fibonacci Heap

Inserción de 65 nodos.

Realizamos un Extract_Min() para que se formen los árboles.



Fibonacci Heap

Vamos a Probar el método Delete(). Sobre el NODO 50 que es el primer hijo del NODO 34 que es el primer hijo del nodo Minimo (NODO 2).

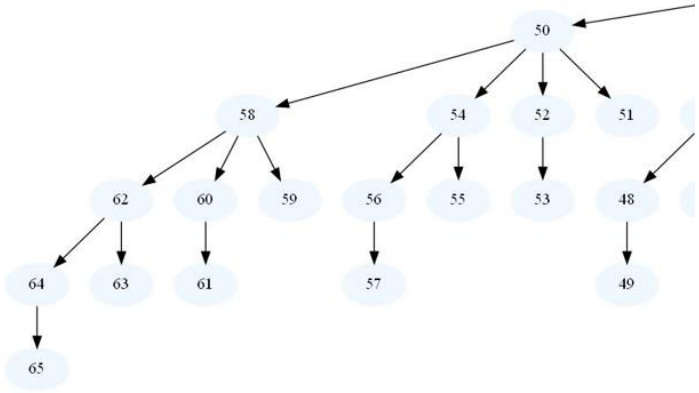
La función Delete() utiliza la función Decrese_Key() y la función Extract_Min().

Así que con eso ya probamos toda la funcionalidad.

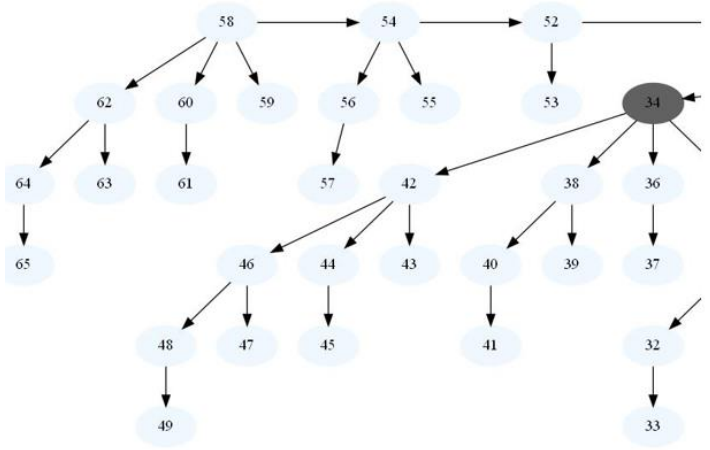
```
NodoF<int> *n = test.GetMin();
NodoF<int> *son = *(n->m_Sons.begin());
NodoF<int> *son_2 = *(son->m_Sons.begin());
test.Delete(son_2);
```

El Nodo que vamos a borrar es el 50, por lo tanto sus hijos van a bajar a las raíces y el nodo padre 34 se pinta de color Grey38.

ANTES



DESPUES



Nota El GraphViz no nos muestra imágenes para una cantidad de nodos muy grande. Sin embargo si podemos generar el .dot.