Colas de Prioridad

Guillermo Palma

Universidad Simón Bolívar
Departamento de Computación y T.I.

CI-2612: Algoritmos y Estructuras II

Plan

Definiciones

Heap

Es una estructura de árbol binario que almacena una colección de claves y tiene las siguientes dos propiedades:

- Todas las hojas en el mismo nivel y todas los nodos internos tienen grado 2, excepto posiblemente por el último nivel, el cual es construido de izquierda a derecha.
- Propiedad del Heap:
  - Para un Max-heap la clave de un nodo x es menor o igual a la clave del padre, esto es $Parent(x) \geq x$
  - Para un Min-heap la clave de un nodo x es mayor o igual a la clave del padre, esto es $Parent(x) \leq x$,

Max-heap

Para todos los nodos x, excepto la raíz, se cumple que $Parent(x) \geq x$

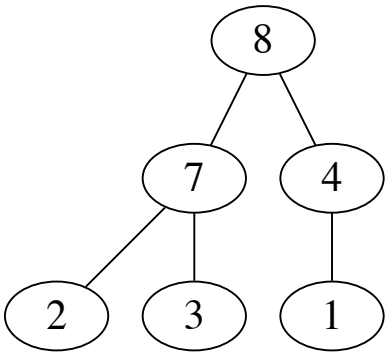

Figura: Ejemplo de un Max-heap

Min-heap

Para todos los nodos x, excepto la raíz, se cumple que $Parent(x) \leq x$

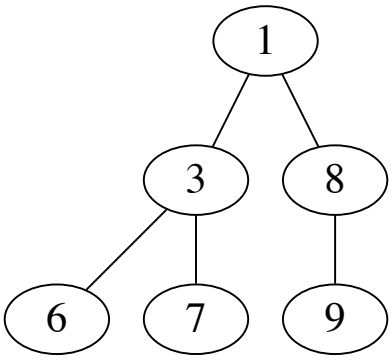

Figura: Ejemplo de un Min-heap

Representación de un Heap

- Se puede representar como un arreglo
- La raíz es $A[1]$
- Padre de $A[i] = A[\lfloor i/2 \rfloor]$ ($Parent\ (i) = \lfloor i/2 \rfloor$)
- Hijo izquierdo de $A[i] = A[2i]$ ($LEFT(i) = 2i$)
- Hijo derecho de $A[i] = A[2i + 1]$ ($RIGHT(i) = 2i + 1$)
- Altura del heap $A \leq length(A)$

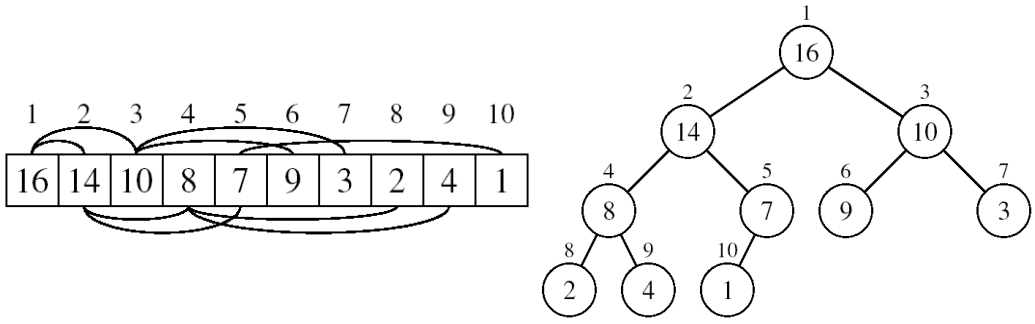

Figura: Ejemplo de un Max-heap. Fuente [1]

MAX-HEAPIFY

- Procedimiento que mantiene las propiedades de un Heap
- Sea un nodo *i* más pequeño que su hijo:
  - Los subárboles izquierdo y derecho de *i* son Max-heaps
  - Intercambia con el hijo más grande
  - Mover la clave hacia bajo del heap
  - Continuar hasta que no haya ningún nodo sea más pequeño que su hijo

Ejemplo de MAX-HEAPIFY

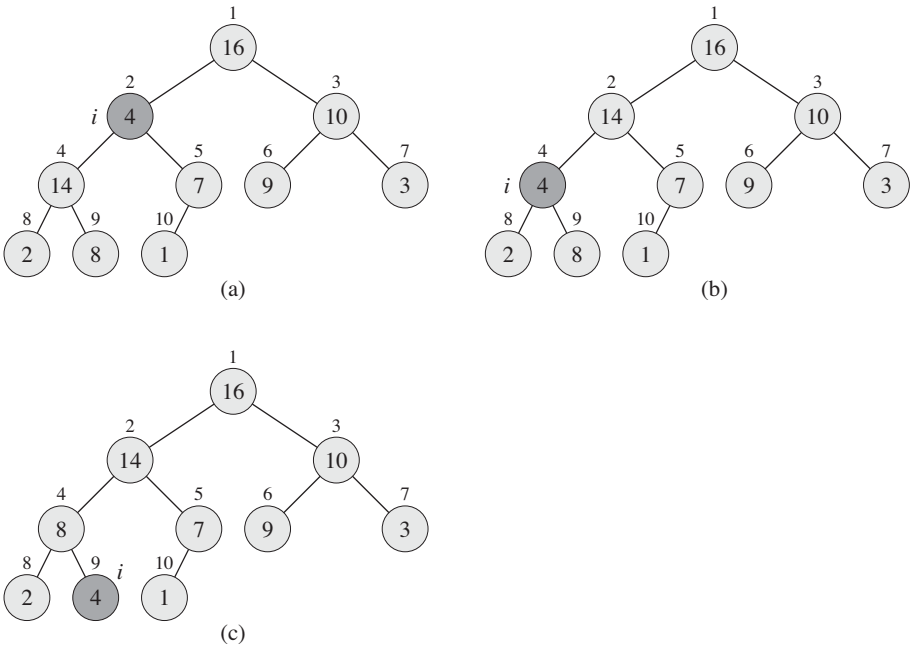

Figura: Llamada `MAX-HEAPIFY(A, 2, 10)`. a) *A*[2] viola la propiedad del Heap. b) *A*[4] viola la propiedad del Heap. c) Se cumple la propiedad del Max-heap. Fuente [1]

Procedimiento MAX-HEAPIFY

Procedimiento MAX-HEAPIFY(A, i, n)

inicio

```
l ← LEFT(i) ;
r ← RIGHT(i) ;
si l ≤ n y A[l] > A[i] entonces
    largest ← l ;
en otro caso
    largest ← i ;
si r ≤ n y A[r] > A[largest] entonces
    largest ← r ;
si largest ≠ i entonces
    SWAP(A[i], A[largest]) ;
    MAX-HEAPIFY(A, largest, n) ;
```

Tiempo del peor caso de MAX-HEAPIFY

- Se recorre el camino más largo de la raíz a la hoja
- En cada nivel se hace dos comparaciones
- O(Altura del heap), esto es $O(\log n)$

Sobre las Colas de Prioridad

- Tipo abstracto de datos que contiene a un conjunto de elementos identificados con una clave, en donde los elementos son requeridos por el orden de sus claves.
- El elemento con más clave más grande (o pequeña) es requerido primero
- Soporta las operaciones de insertar, eliminar el máximo (mínimo), obtener el máximo (mínimo), e incrementar clave.
- Ejemplos de usos de las Colas de Prioridad:
  - El planinificador de procesos de un OS, tiene una cola de prioridad para permitir el acceso al CPU al proceso de mayor prioridad
  - Se usan en los algoritmos para determinar el árbol mínimo cobertor
  - Se usan en los algoritmos para determinar caminos de costo mínimo
- Posible implementaciones:
  - Como un arreglo
  - Como una lista enlazada
  - Como un Max-Heap (o Min-Heap)

Operaciones de las Colas de Prioridad

Dada una representación de una Cola de Prioridad como un conjunto ordenado S, en donde nos interesa obtener el elemento mayor de S se tienen las siguientes operaciones:

- INSERT(S, x): Incluye un elemento con clave x en el conjunto S
- MAXIMUM(S): Obtiene el elemento x con la clave más grande
- INCREASE-KEY(S, x, k): Incrementa la clave del x en el conjunto S, con la nueva clave k
- EXTRACT-MAX(S): Elimina el elemento con la clave más grande de S

Vamos a suponer que se implementa una Cola de Prioridad como un Max-Heap.

HEAP-MAXIMUM

Función HEAP-MAXIMUM(A)

inicio
  retornar $A[1]$

HEAP-MAXIMUM es $O(1)$

Ejemplo de HEAP-MAXIMUM

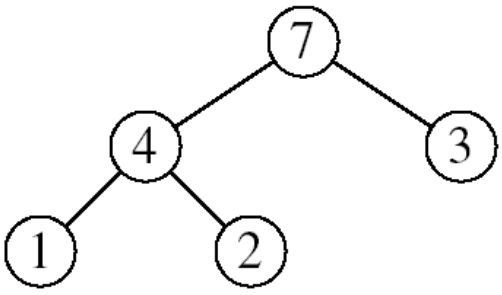

Figura: En este Max-Heap se tiene que HEAP-MAXIMUM retorna 7

HEAP-EXTRACT-MAX

- Extrae el elemento con clave más grande del Max-Heap
- Intercambia el elemento raíz con el último elemento
- Se decrementa el tamaño del Max-Heap
- Se llama a MAX-HEAPIFY para arreglar el valor de la nueva raíz

Ejemplo de HEAP-EXTRACT-MAX

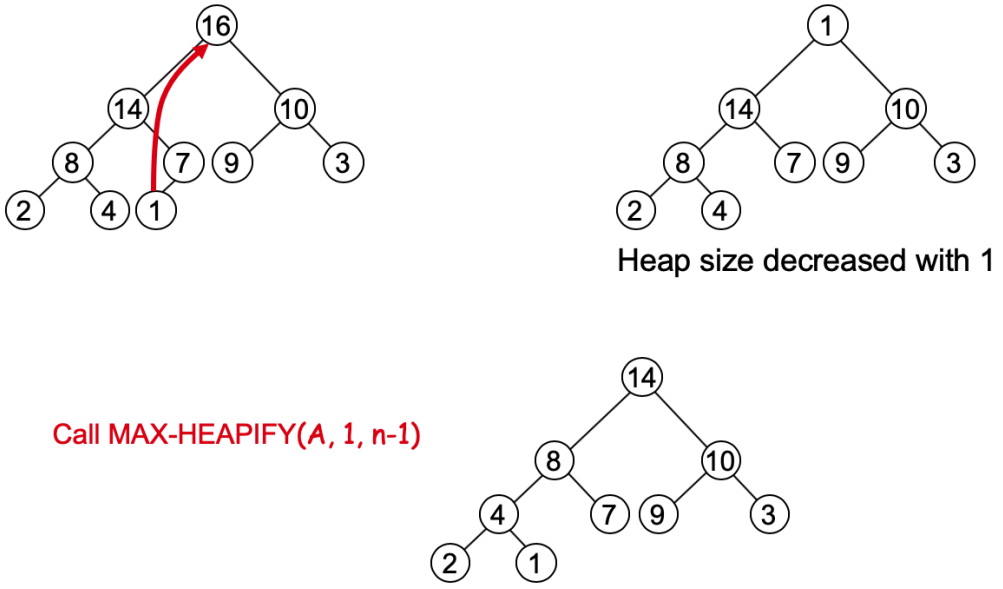

Figura: Se extrae el elemento 16 del Max-Heap

Función HEAP-EXTRACT-MAX

Función HEAP-EXTRACT-MAX(A)

inicio

si $A.heapSize < 1$ **entonces**

retornar error heap underflow

$max \leftarrow A[1]$;

$A[1] \leftarrow A[A.heapSize]$;

$A.heapSize \leftarrow A.heapSize - 1$;

`MAX-HEAPIFY(`$A, 1$`)` ;

retornar *max*

HEAP-EXTRACT-MAX es $O(\log n)$

HEAP-INCREASE-KEY

- Se incrementa una clave existente en el conjunto
- Se chequea si la nueva clave viola las propiedades del heap.
- Si se violan entonces se atraviesa el árbol hasta la raíz o hasta encontrar la posición correcta para la nueva clave

Ejemplo de HEAP-INCREASE-KEY

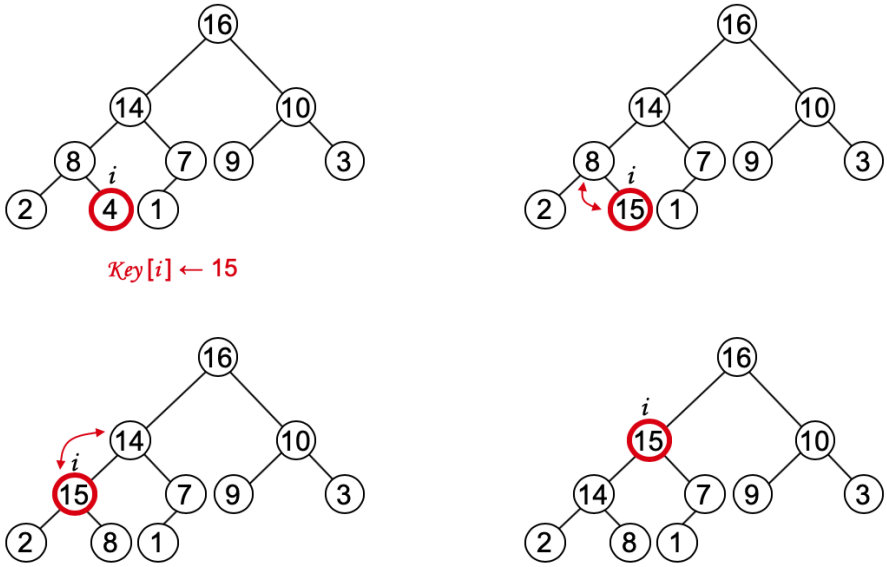

Figura: Se incrementa la clave 4 a 15 en el Max-Heap

Procedimiento HEAP-INCREASE-KEY

Procedimiento HEAP-INCREASE-KEY(A, i, key)

inicio

- **si** $key < A[i]$ **entonces**
  - error la nueva clave es menor
- $A[i] \leftarrow key$;
- **mientras** $i > 1 \wedge A[PARENT(I)] < A[i]$ **hacer**
  - SWAP $(A[i], A[PARENT(i)])$;
  - $i \leftarrow PARENT(i)$;

HEAP-INCREASE-KEY es $O(\log n)$

MAX-HEAP-INSERT

- Se quiere insertar un elemento nuevo en la Cola de Prioridad
- Se aumenta el tamaño del Max-Heap en una unidad
- Se agrega al final del Max-Heap una hoja con clave menos infinito
- Se usa el procedimiento HEAP-INCREASE-KEY para incrementar la clave con menos infinito, con el valor que se quiere insertar

Ejemplo de MAX-HEAP-INSERT

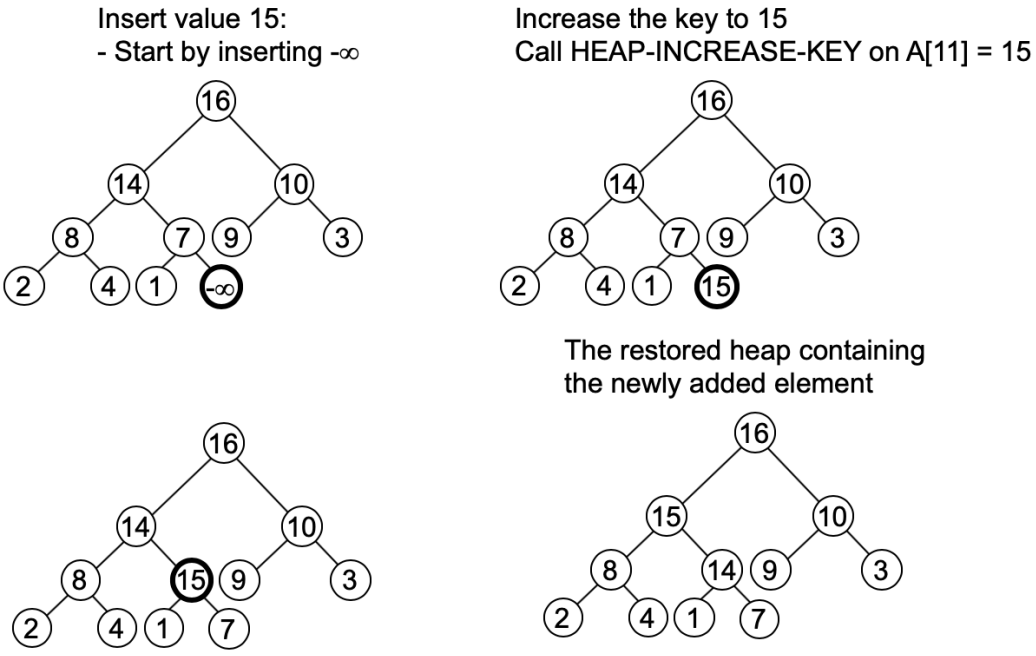

Figura: Se inserta la clave 15 en el Max-Heap

Procedimiento MAX-HEAP-INSERT

Procedimiento MAX-HEAP-INSERT(A, key)

inicio

$A.heapSize \leftarrow A.heapSize + 1$;
$A[A.heapSize] \leftarrow infinitoNegativo$;
HEAP-INCREASE-KEY(A, $A.heapSize$, key)

MAX-HEAP-INSERT es $O(\log n)$

Referencias

T. Cormen, C. Leirserson, R. Rivest, and C. Stein.
Introduction to Algorithms.
McGraw Hill, 3ra edition, 2009.