Heaps

Ernesto Rodriguez - Juan Roberto Alvaro Saravia

Universidad Francisco Marroquin

ernestorodriguez@ufm.edu - juanalvarado@ufm.edu

Heap Binario

- Se puede representar con un arreglo
- Cada elemento del arreglo es un nodo del heap
- Todo elemento (menos el primero) tiene un padre
- Todo elemento tiene como maximo dos hijos: derecho e izquierdo
- Existe una porpiedad p, tal que:

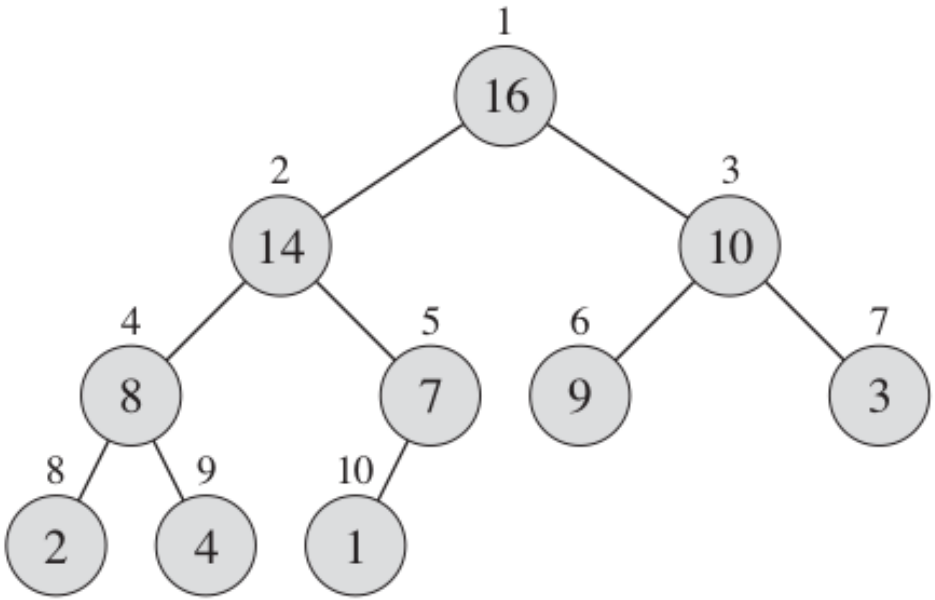
$$\forall\ n_i \in Ns.\ p\ (\texttt{padre}\ n_i)\ n_i$$

- La propiedad p define un orden: $p\ a\ b\ \&\ p\ b\ c \rightarrow\ p\ a\ c$
- Cuando un heap se representa con un arreglo:

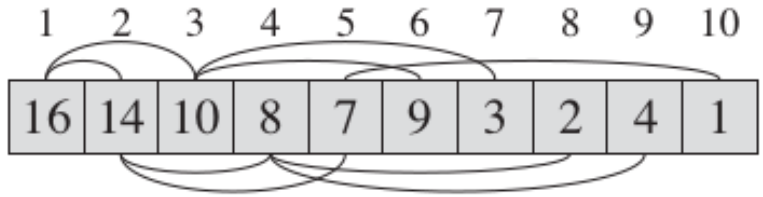
$$\begin{aligned}
&\texttt{padre}\ n_i := n_{i/2} \\
&\texttt{hijo_izquierdo}\ n_i := n_{2i} \\
&\texttt{hijo_derecho}\ n_i := n_{2i+1}
\end{aligned}$$

Heap Binario

Si $p\ a\ b := a \geq b$



(a)



(b)

Min y Max Heap

- En este curso se consideraran dos heaps:
  - Cuando (max heap) $p\ a\ b := a \geq b$
  - Cuando (min heap) $p\ a\ b := a \leq b$
- ¿Que valor seria la raiz en un min y max heap?
- ¿Que caracteristica cumplen los hijos de todo valor?
- Los algoritmos de ordenamiento por lo general utilizan el max heap
- Un min heap se utiliza a menudo para colas
- **Nota:** A partir de este momento, la palabra heap se referira a un max heap.

La invariante de heaps

- ¿Que propiedades se cumplen para todo heap?

La invariante de (max) heaps

- ¿Que propiedades se cumplen para todo heap?
  1. $\forall\ n_i \in Ns.\ \texttt{padre}\ n_i \geq n_i$
  2. $\forall\ n_i \in Ns.\ \texttt{hijo\_derecho}\ n_i \leq n_i$
  3. $\forall\ n_i \in Ns.\ \texttt{hijo\_izquierdo}\ n_i \leq n_i$
- Se utilizara la primera (1) como la invariante. Tambien se llamara la "Condicion de un heap".

Heapify

- Algoritmo que restaura la condicion de un heap.
- Funciona inductivamente:
  - Dado un elemento n_i, asumir que los heaps encabezados por "`hijo_derecho` n_i" e "`hijo_izquierdo` n_i" son un heap y cumplen la condición de heap.

Heapify

- Algoritmo que restaura la condicion de un heap.
- Funciona inductivamente:
  - Dado un elemento n_i, asumir que los heaps encabezados por "`hijo_derecho` n_i" e "`hijo_izquierdo` n_i" son un heap y cumplen la condición de heap.
  - Si $n_i <$ `hijo_derecho` n_i o $n_i <$ `hijo_izquierdo`, intercambiar el mayor de estos valores con n_i
  - Repetir el algoritmo con el valor que fue intercambiado.

Heapify

Algorithm 1 Heapify

```
1:  procedure HEAPIFY(Ns, i)
2:      izq ← hijo_izquierdo(i)
3:      der ← hijo_derecho(i)
4:      mayor ← i
5:      if izq ≤ len(Ns) and Ns[izq] > Ns[mayor] then
6:          mayor ← izq
7:      if der ≤ len(Ns) and Ns[der] > Ns[mayor] then
8:          mayor ← der
9:      if mayor ≠ i then
10:         intercambiar(Ns, i, mayor)
11:         heapify(Ns, mayor)
```