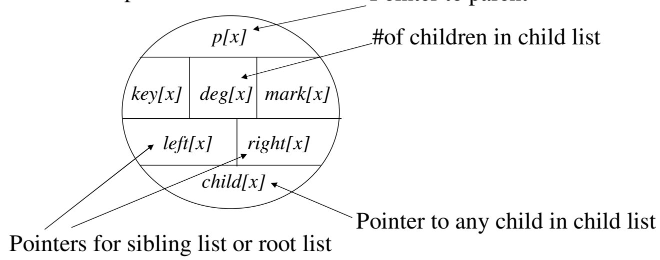
Fibonacci Heaps

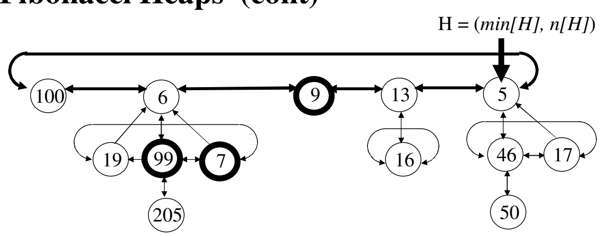
A Fibonacci Heap is an unordered circular doubly linked

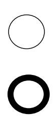
list of heap-ordered trees.



mark[x] = # of children *x* has lost since *x* was last adopted
(will always be 0 or 1)

Fibonacci Heaps (cont)





n[H] = # of nodes in H

t[H] = # of trees in H

D(n) = maximum degree possible in any Fibonacci heap with *n* nodes

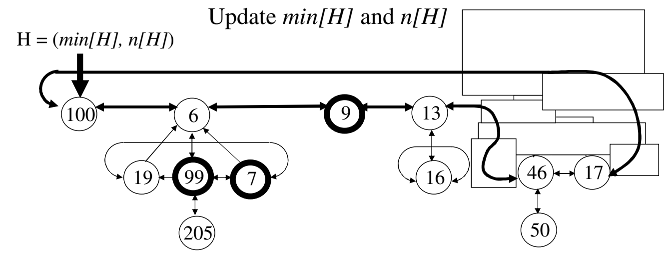
Fibonacci Heaps (cont)

Operations

$Make-Heap()$	$min[H] \leftarrow nil \quad n[H] \leftarrow 0$	$\Theta(1)$
$Minimum(H)$	Return *key[min[H]]*.	$\Theta(1)$
$Insert(H, x)$	Add new single node to list and update *min[H]* and *n[H]*	$\Theta(1)$
$Union(H_1, H_2)$	Concatenate root lists and update *min[H]* and *n[H]*	$\Theta(1)$
$Delete(H, x)$	Decrease key of x to -∞ and then delete minimum	?

Fibonacci Heaps (cont)

$Delete-Min(H)$
Remove *min[H]* from root list
Add *min[H]* 's children to root list
`Consolidate-Heap`
Update *min[H]* and *n[H]*



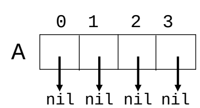
Fibonacci Heaps (cont)

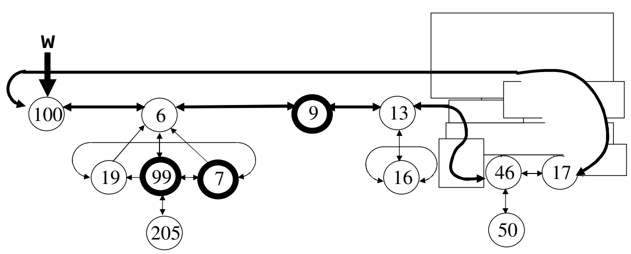
Consolidate-Heap

```
for each root w in root-list
    remove w; d ← degree(w)
    while(A[d] ≠ nil)
        w ← Link(w,A[d])
        A[d] ← nil; d ← d+1
    endwhile
    A[d] ← w
endfor
for d ← 0 to D(n)
   If A[d] ≠ nil then
       Append A[d] to root-list
endfor
```


Fibonacci Heaps (cont)

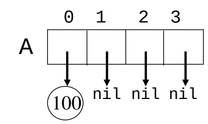
Consolidate-Heap

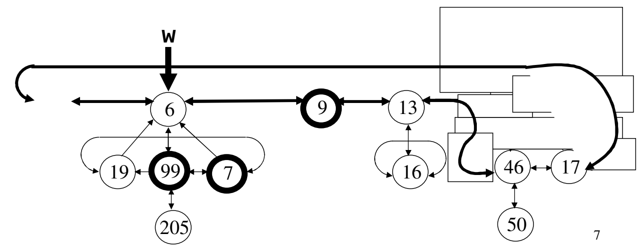




Fibonacci Heaps (cont)

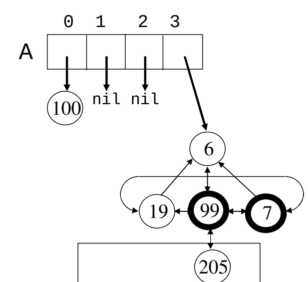
Consolidate-Heap

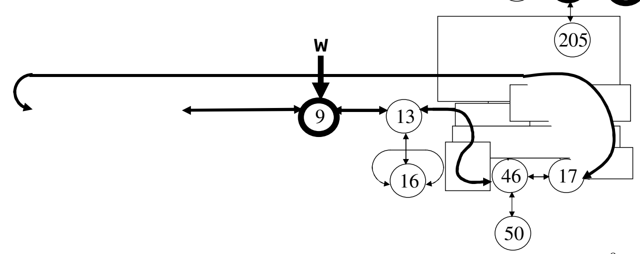




Fibonacci Heaps (cont)

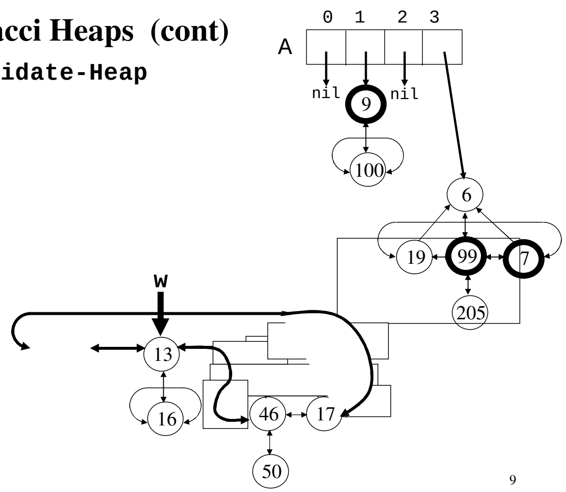
Consolidate-Heap





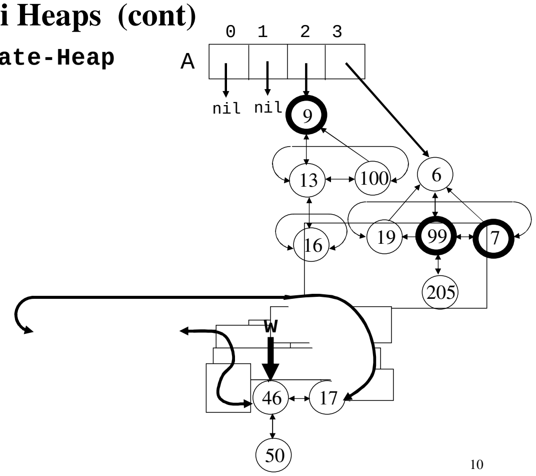
Fibonacci Heaps (cont)

Consolidate-Heap



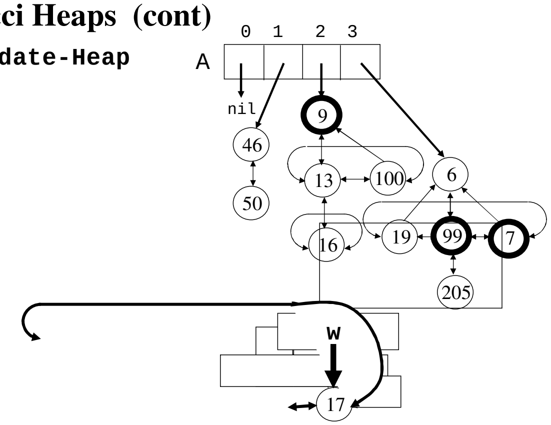
Fibonacci Heaps (cont)

Consolidate-Heap



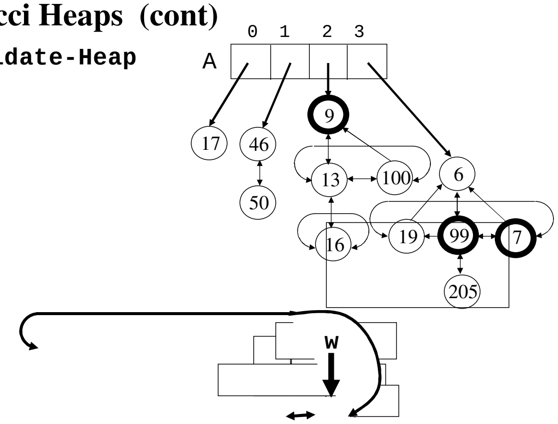
Fibonacci Heaps (cont)

Consolidate-Heap



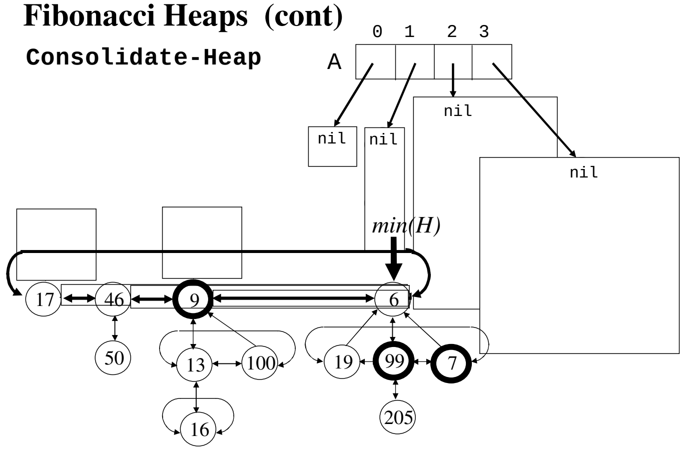
Fibonacci Heaps (cont)

Consolidate-Heap



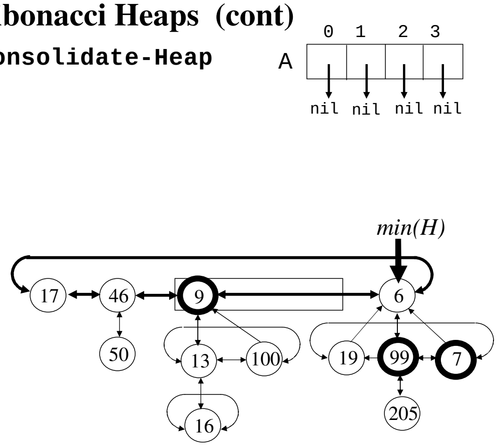
Fibonacci Heaps (cont)

`Consolidate-Heap`



Fibonacci Heaps (cont)

`Consolidate-Heap`



Fibonacci Heaps (cont)

`Consolidate-Heap` takes time $O(D(n)+t(H))$

Need to know D(n) to make buckets

\# buckets

\# trees before > #trees linked

$D(n) = O(\lg n)$ (to be shown)

Worst case: $\Omega(n)$

Without Decrease or Delete, no nodes will be marked and trees will be unordered binomial trees

Fibonacci Heaps (cont)

$Decrease(H, x, k)$

`key[x]`←`k`

Traverse heap up from `x` to first unmarked ancestor.

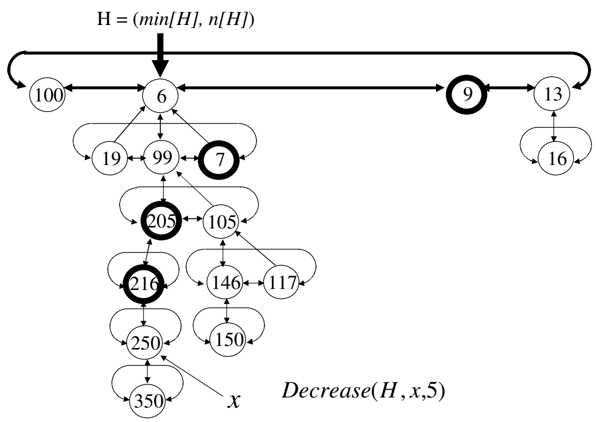
Mark this ancestor.

Promote `x` and any traversed ancestors to the root-list and unmark them.

Update *min(H)*

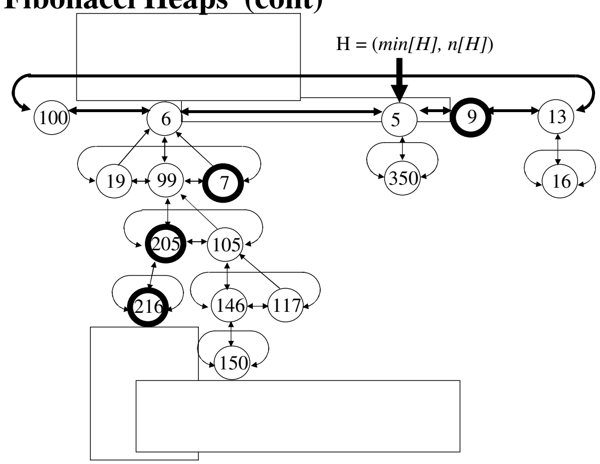
Decrease takes time $O(c)$ where c is the number of nodes promoted. **Worst case:** $\Omega(n)$

Fibonacci Heaps (cont)

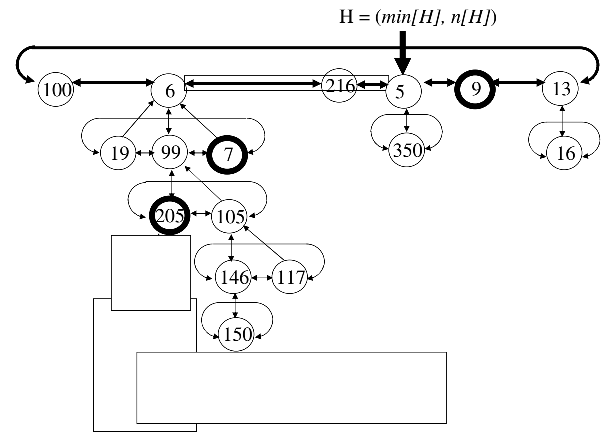


$Decrease(H, x, 5)$

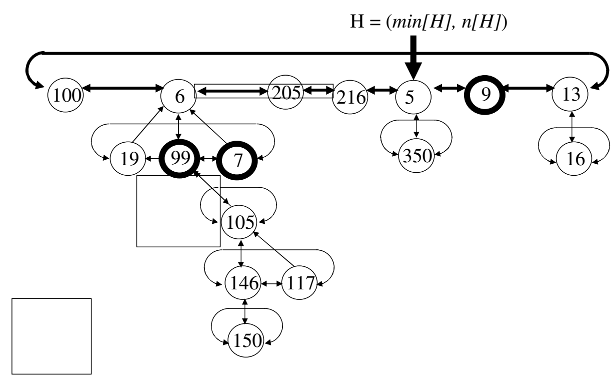
Fibonacci Heaps (cont)



Fibonacci Heaps (cont)



Fibonacci Heaps (cont)



Fibonacci Heaps (cont)

Lemma If $y_1, y_2, \ldots, y_k$ are the children of node x in order of longevity (order of adoption) then for $2 \le i \le k$ $\deg(y_i) \ge i-2$.

Proof: Node y_i became a child of x only in `Consolidate`

Node x already had children $y_1, y_2, \ldots, y_{i-1}$ at that point.

Node x and node y_i had the same degree when y_i was adopted.

Node y_i had degree at least *i-1* when it was adopted.

Node y_i has lost at most one child since being adopted.

Node y_i has degree at least *i-2*.

QED

Fibonacci Heaps (cont)

Lemma $F_{k+2} = 1 + \sum_{i=0}^{k} F_i$ for $k \ge 0$

Let $size(x)$ = # nodes in x's subtree .

Lemma If $\deg(x) = k$ then $size(x) \ge F_{k+2}$.

Proof:

Let $s_k = \min\{size(x) | \forall\, x \text{ with degree } k \text{ in any Fibonacci heap}\}$.

Node x has k children of degrees at least *-1 ,0, 1, 2, k-2*.

$$size(x) \ge s_k = 1 + \sum_{i=1}^{k} s_{i-2} \ge 2 + \sum_{i=2}^{k} s_{i-2} \ .$$

Fibonacci Heaps (cont)

Claim: $s_k \ge F_{k+2}$.

Base: $k=0 \quad s_o = 1 \quad F_2 = 1$

$k=1 \quad s_1 = 2 \quad F_3 = 2$

Step: $k \ge 2$

$$s_k \ge 2 + \sum_{i=2}^{k} s_{i-2} \ge 2 + \sum_{i=2}^{k} F_i = 1 + \sum_{i=0}^{k} F_i = F_{k+2}$$

So $size(x) \ge s_k \ge F_{k+2}$ **QED**

Corollary $D(n)$ is $O(\lg n)$

$$n \ge size(x) \ge F_{k+2} \ge \left(\frac{1+\sqrt{5}}{2}\right)^k$$

So $\deg(x) = k \le \dfrac{\lg n}{\lg \frac{1+\sqrt{5}}{2}} = O(\lg n)$

Fibonacci Heaps (cont)