Binomial queues

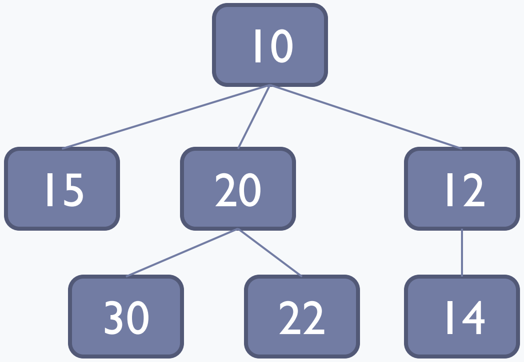
INFO1905 Advanced topic

Priority queues and heaps

- Priority queues are used very often in algorithm implementations
  - Examples: shortest paths, spanning trees, encoding algorithms, scheduling problems, bipartite matching problems, network flow and many more
- Main operations in the abstract data type of a priority queue:
  - Insert(key, value)
  - FindMin( )
  - DeleteMin( )

Priority queue implementations

- Heaps
  - We have discussed binary heaps
- Heap ordered trees
  - All non-root nodes follow the heap-order:
    - The key is greater than the key of its parent

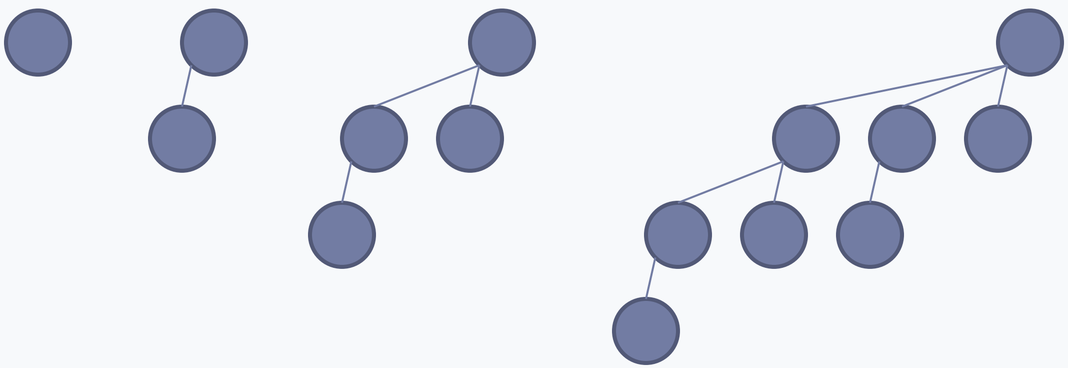


Binomial queues

- Heaps are simple and very efficient…
  - But merging two heaps in one is not so efficient
- A binomial queue is an efficient priority queue implementation that also allows fast merging
  - Also called "melding"

Binomial trees

- Binomial trees have the following structure
  - (B_0, B_1, B_2, B_3 are shown below)

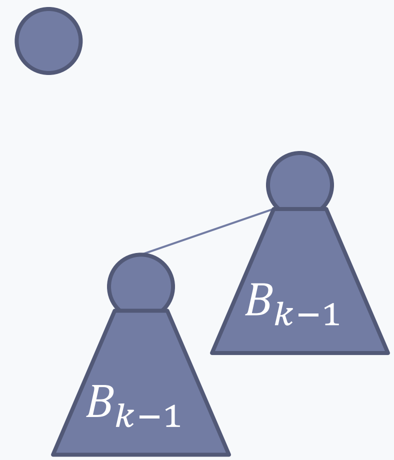


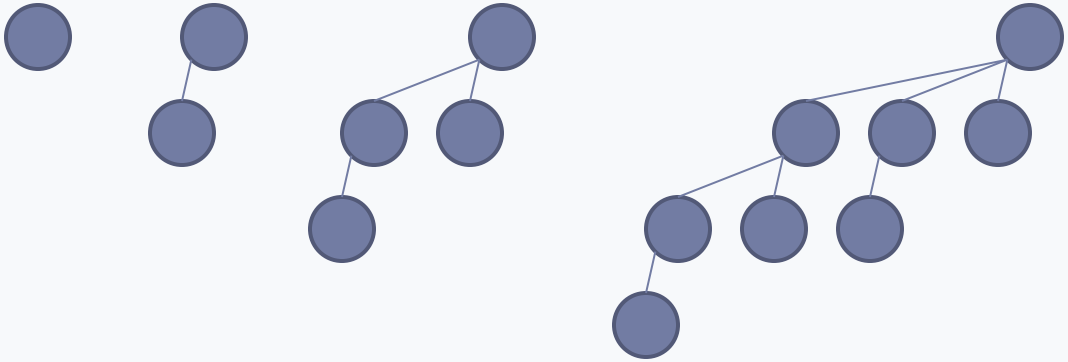
Binomial trees

- Binomial trees

B_0:

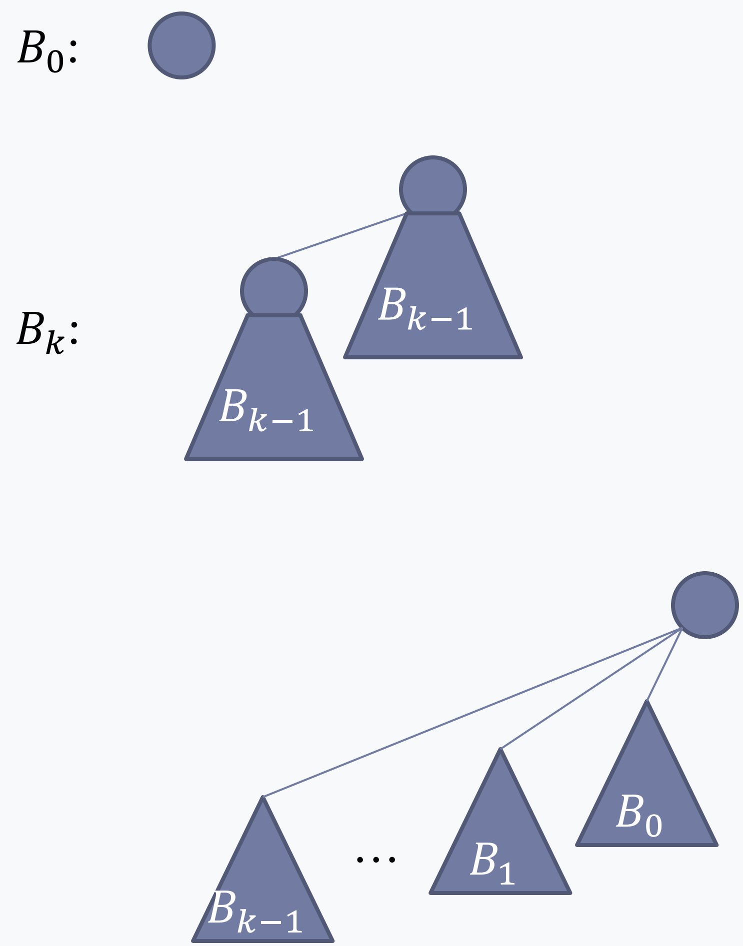
B_k:





Binomial trees

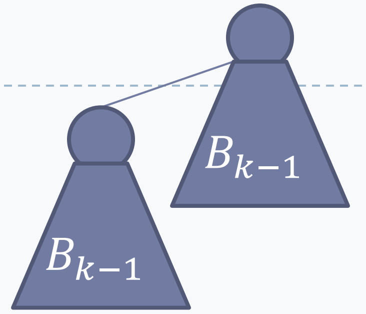
- Binomial trees

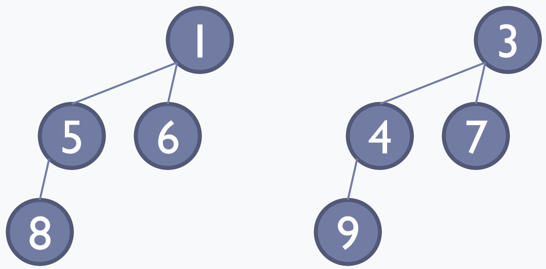


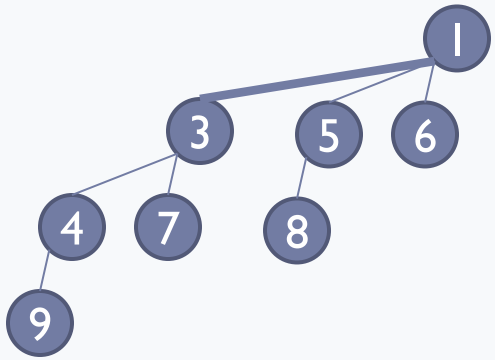
Binomial trees - merging

- Merging two heap ordered binomial trees is easy

B_k:



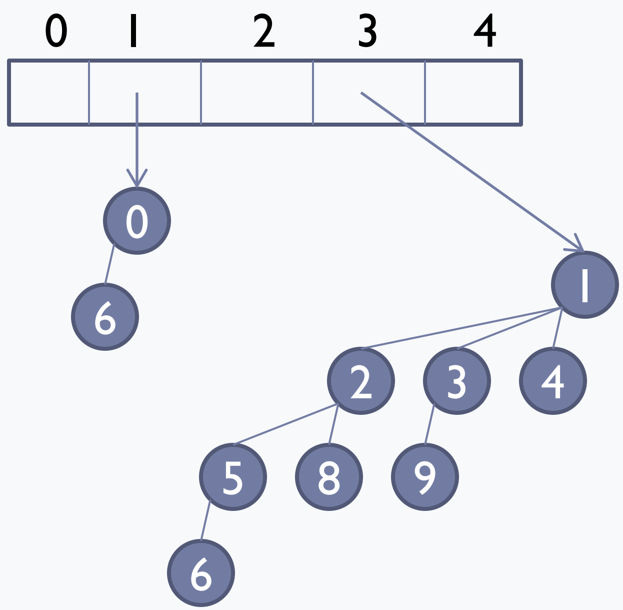




 Advanced material

Binomial queue

- A binomial is a collection of heap ordered binomial trees
- A binomial queue never contains two B_k trees for any k

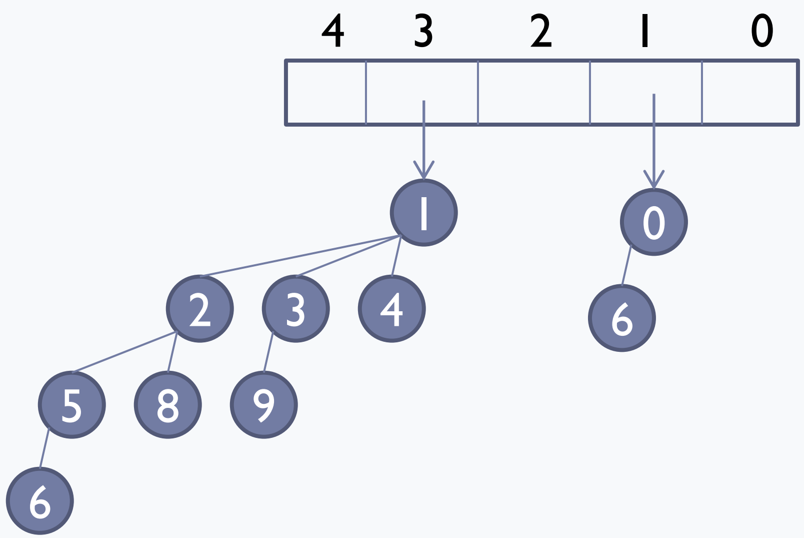


This binomial queue has 10 elements

10 in binary is 1010.
The binomial queue will have a binomial tree B_1 and a B_3

Binomial queue

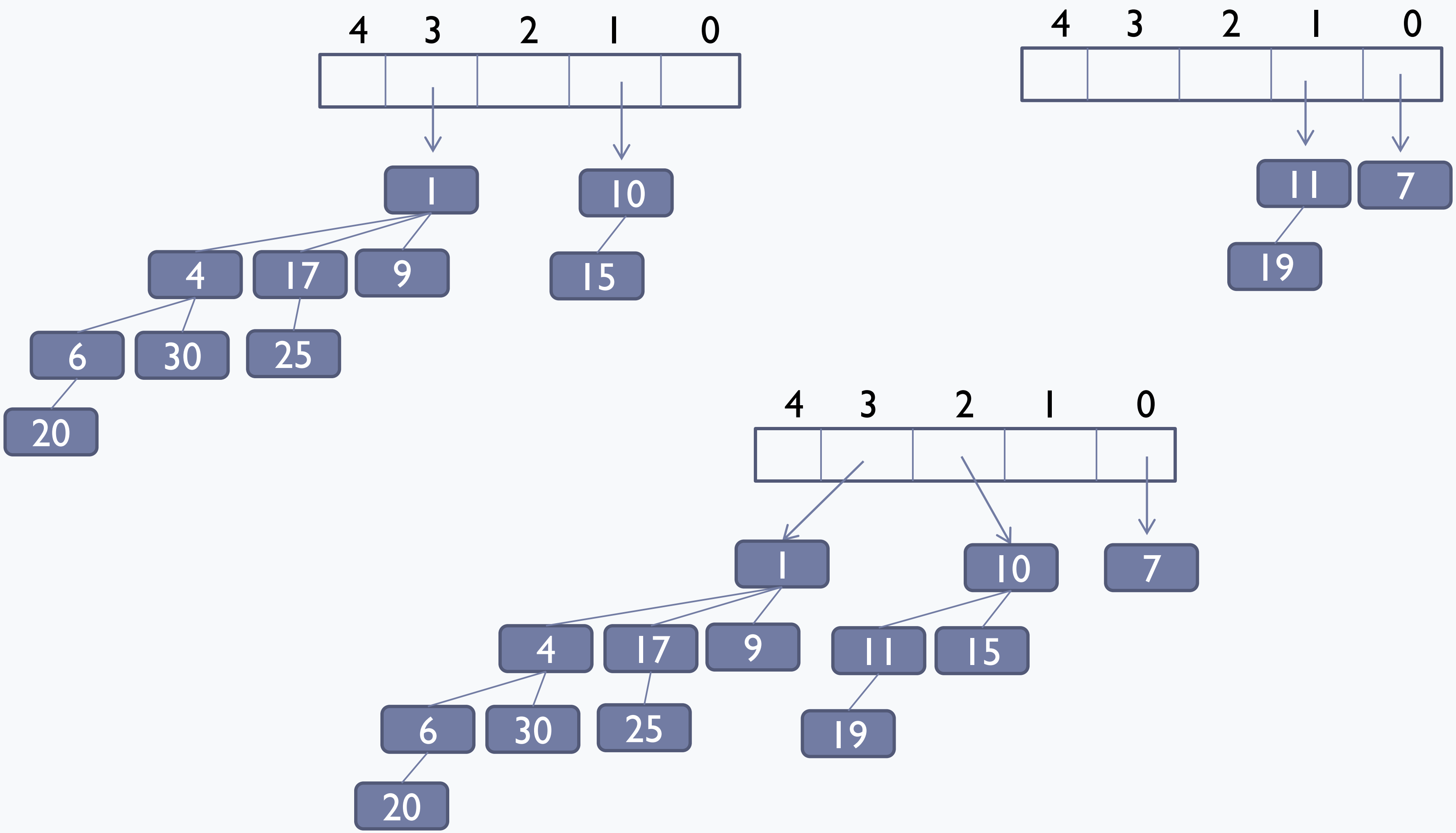
- A binomial is a collection of heap ordered binomial trees
- A binomial queue never contains two B_k trees for any k



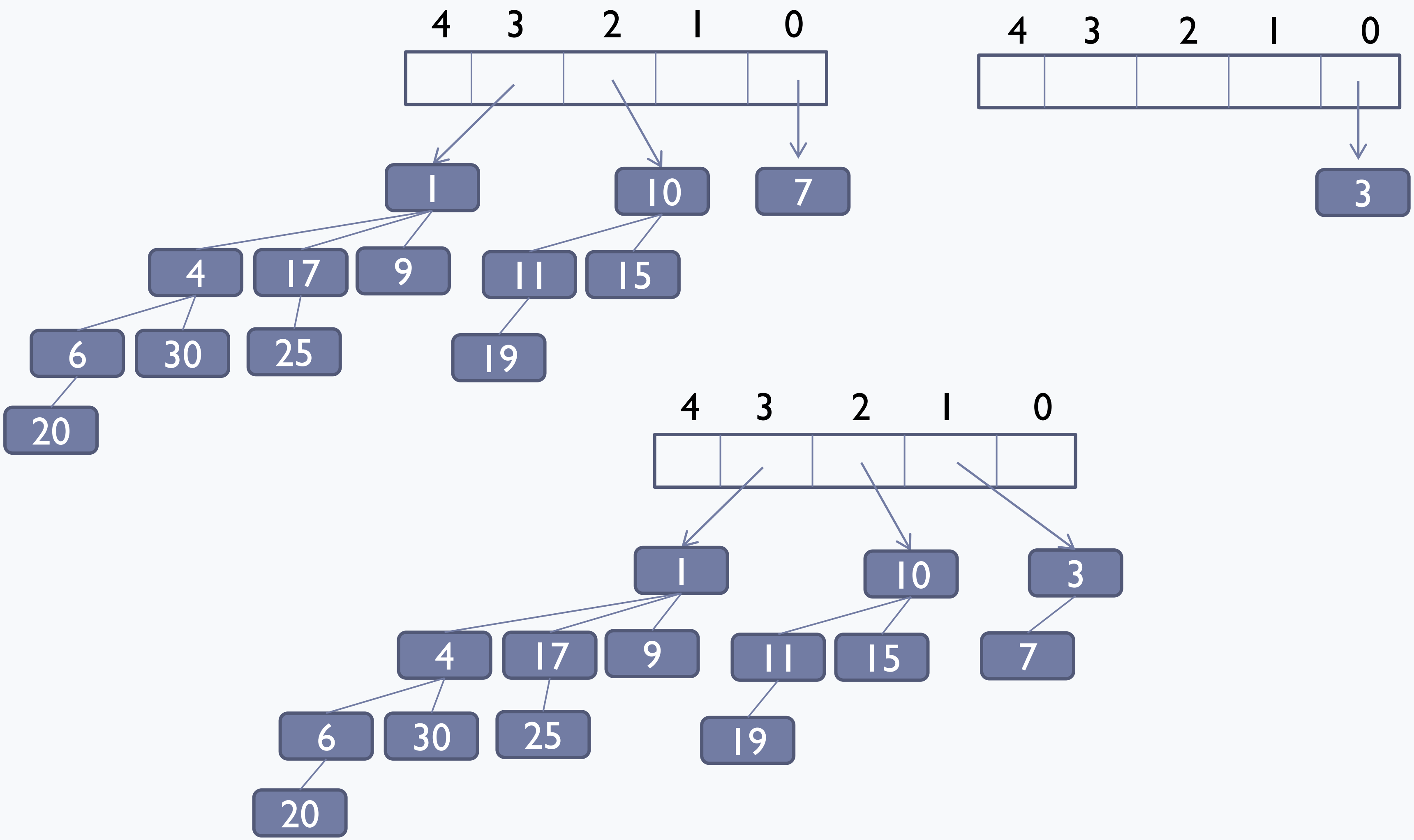
This binomial queue has 10 elements

10 in binary is 1010.
The binomial queue will have a binomial tree B_1 and a B_3

Binomial queue operations

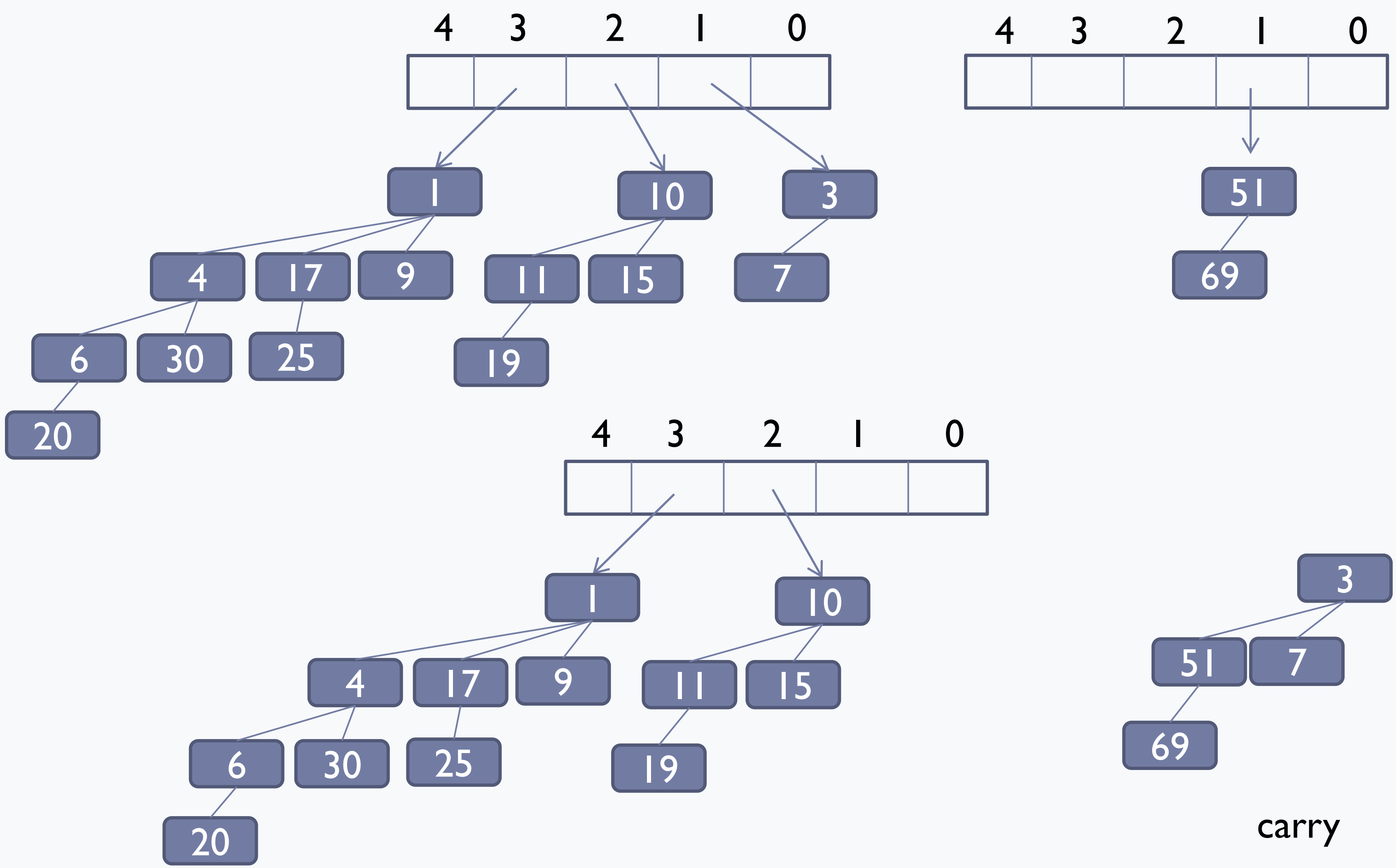


Binomial queue operations - merge



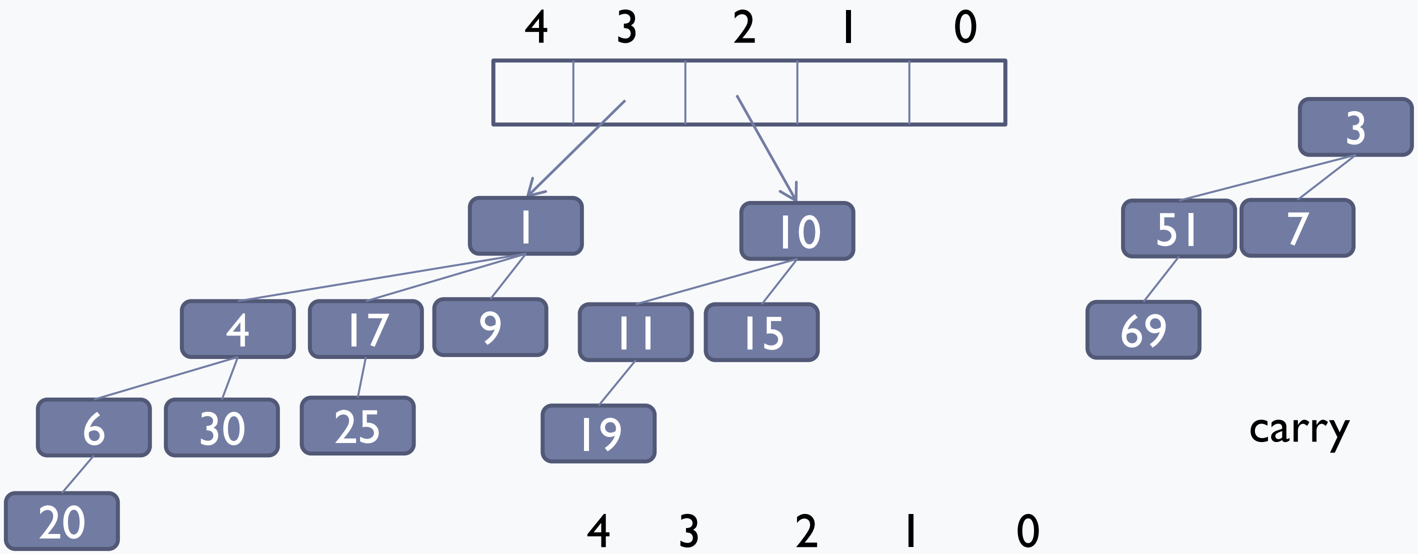
 Advanced material

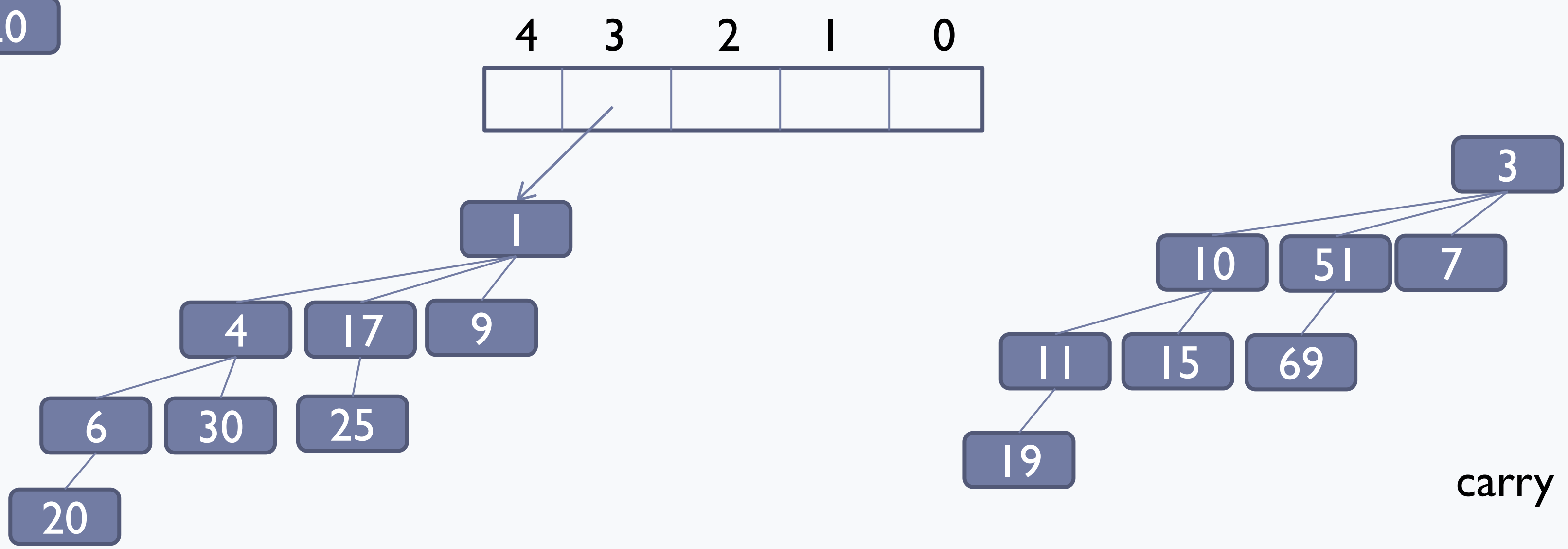
Binomial queue operations - merge



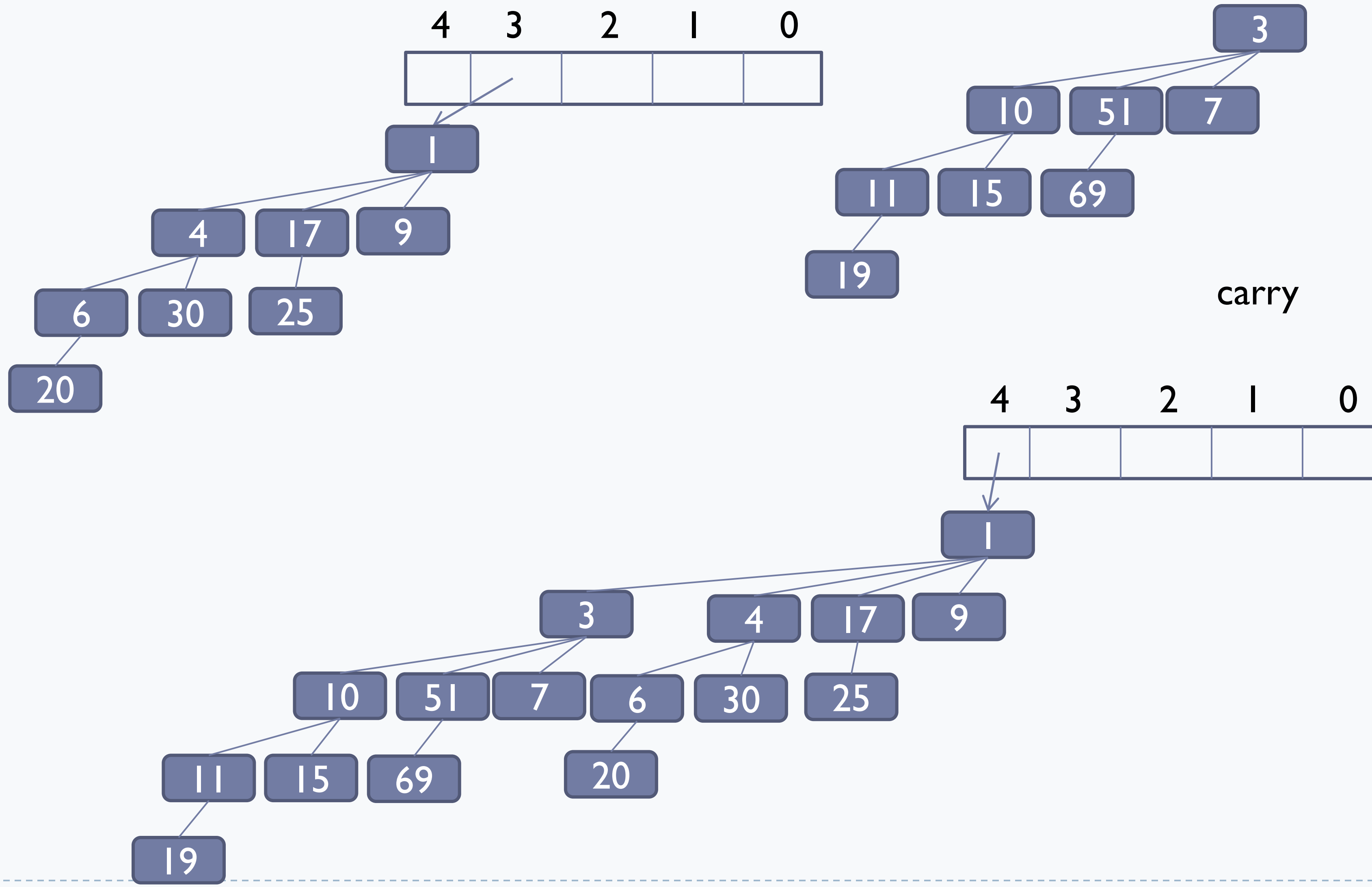
 Advanced material

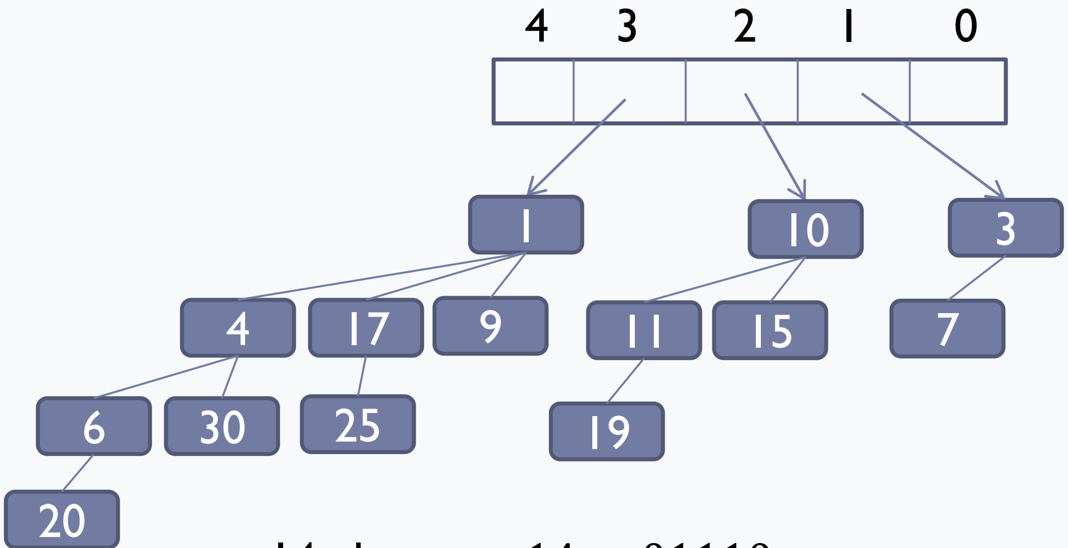
Binomial queue operations - merge



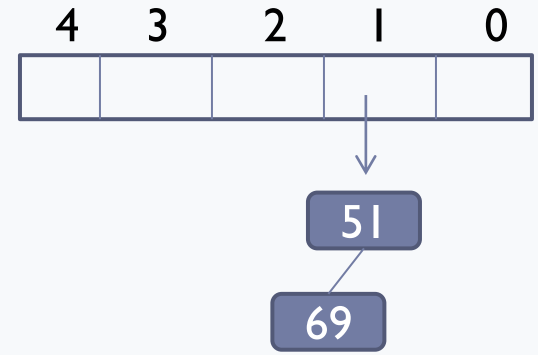


Binomial queue operations - merge

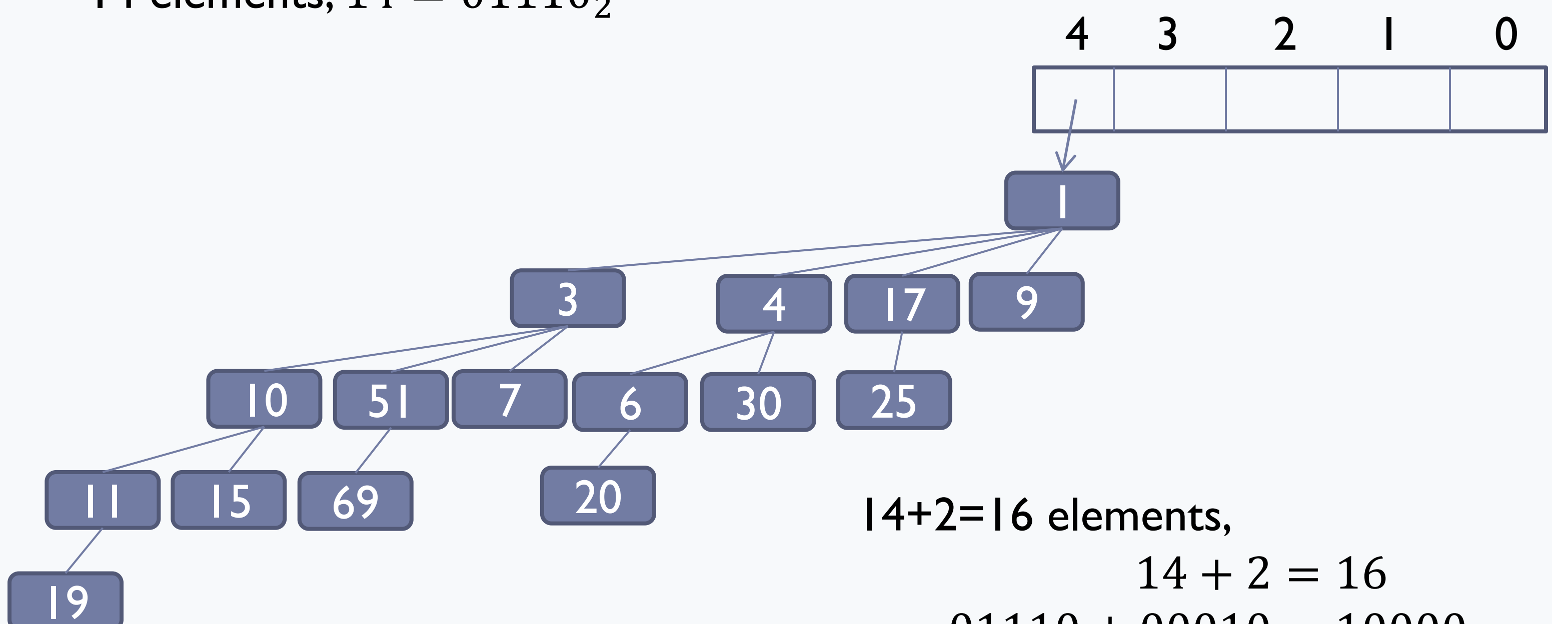




14 elements, $14 = 01110_2$



2 elements, $2 = 00010_2$



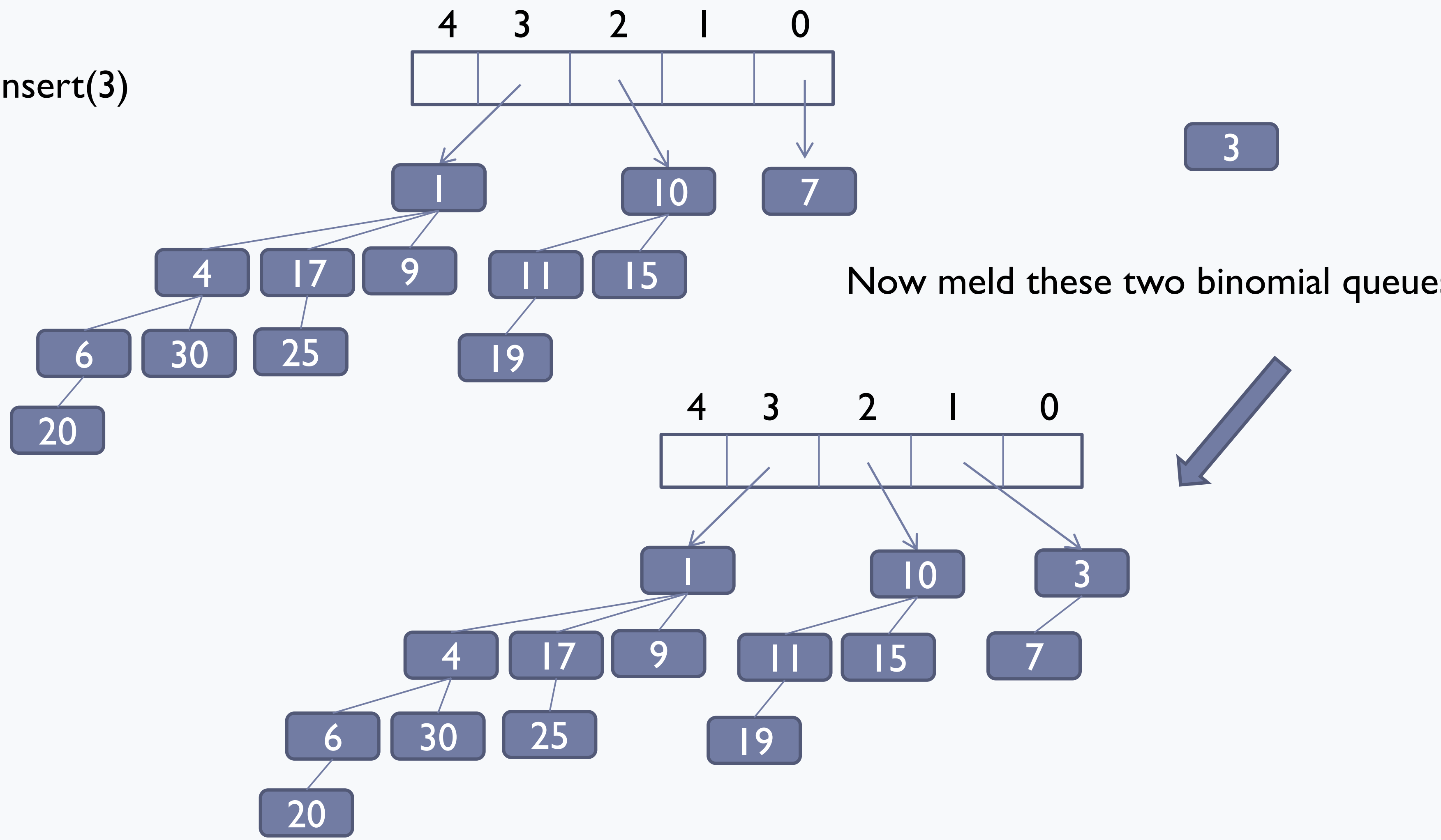
14+2=16 elements,

$$14 + 2 = 16$$
$$01110 + 00010 = 10000$$

 Advanced material

Binomial queue operations - insert

Insert(3)



Binomial queue facts

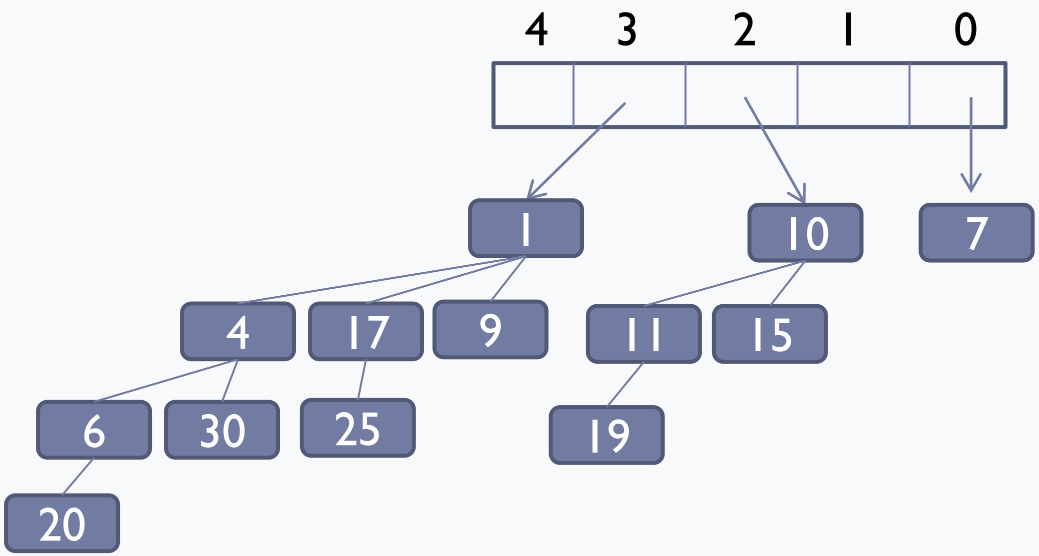
- Number of nodes in B_r
- $$\begin{aligned} size(B_r) &= 2 \cdot size(B_{r-1}) \\ &= 2 \cdot 2 \cdot size(B_{r-2}) \\ &= \cdots \\ &= 2^r \end{aligned}$$
- Height of B_r is $r + 1$
- Largest tree possible in a binomial queue with n elements B_r with $r = \log n$
  - A larger tree would have more than n nodes by itself
- Maximum number of trees in a binomial queue with n elements is $\log n$
  - A binomial queue never contains two trees of the same order

Advanced material

Binomial queues: merge

- Worst case complexity
- At most one link per tree merge
- At most one tree merge per position
- At most $\log n$ trees in the queue
- Overall $O(\log n)$ worst case

Binomial queue operations – find minimum

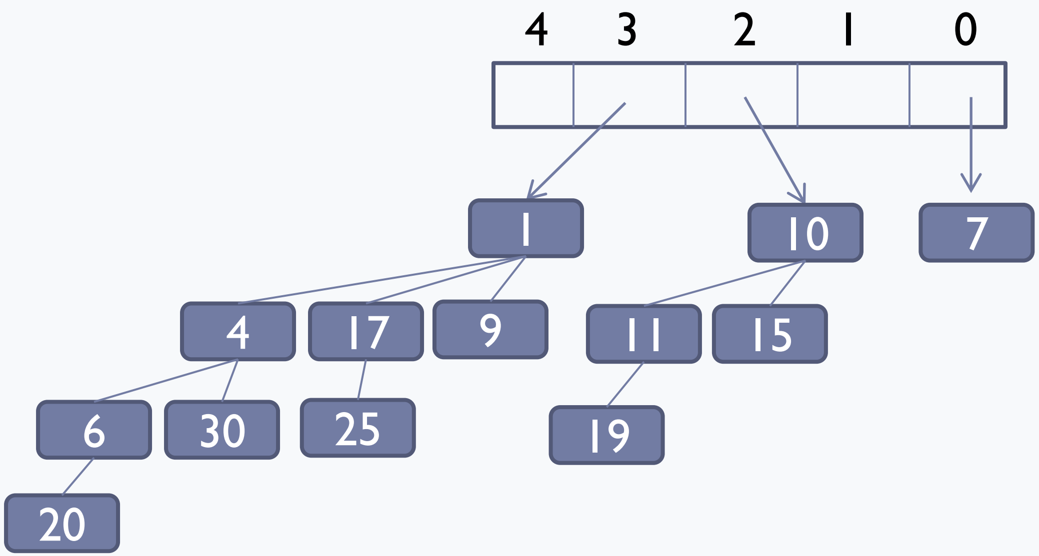


Go through all trees, and find the one with the smallest element at the root

Worst case complexity: $O(\log n)$

 Advanced material

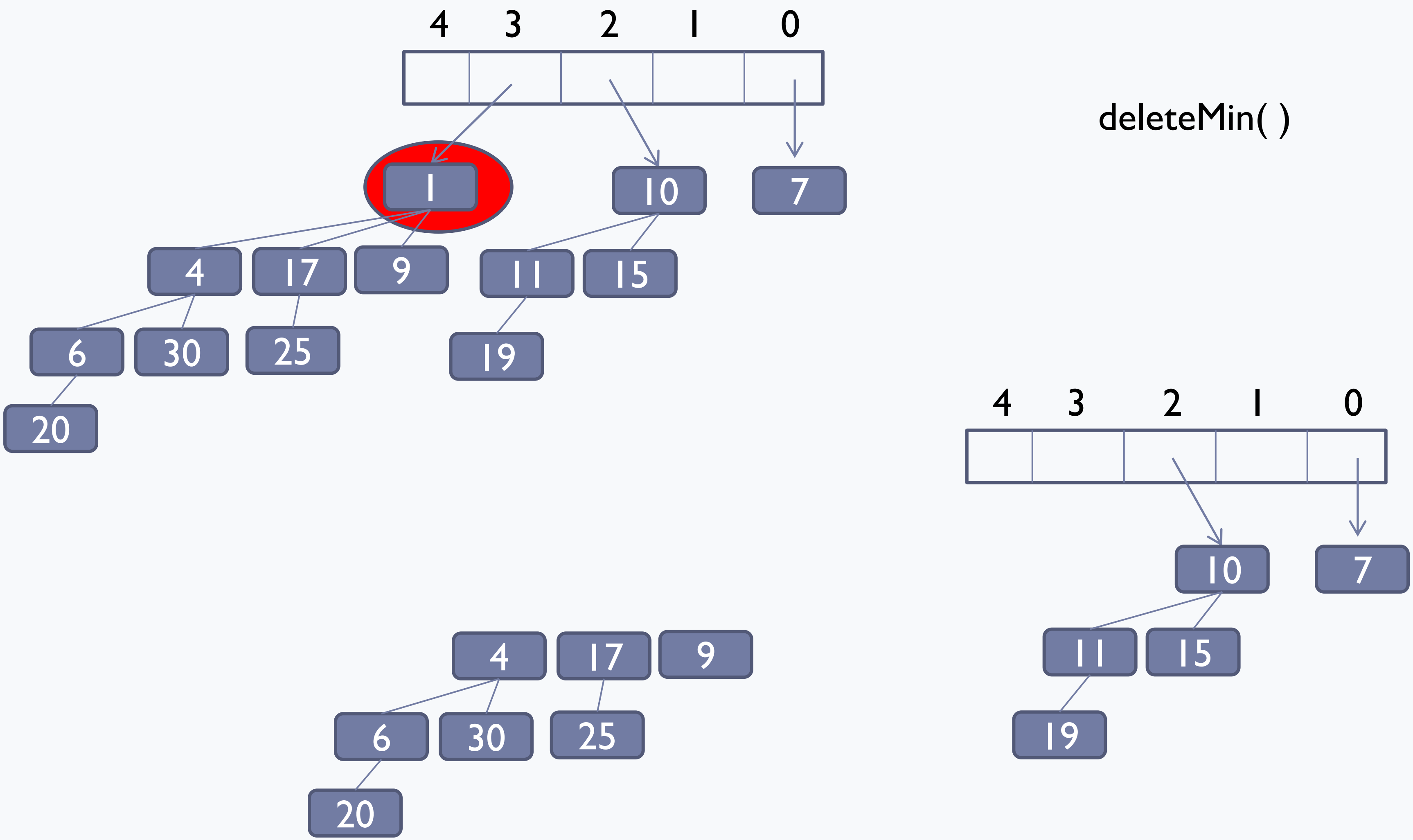
Binomial queue operations – delete minimum



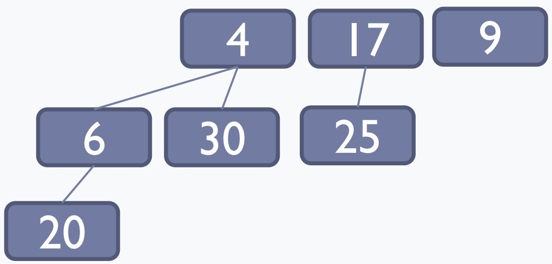
Go through all trees, and find the one with the smallest element at the root

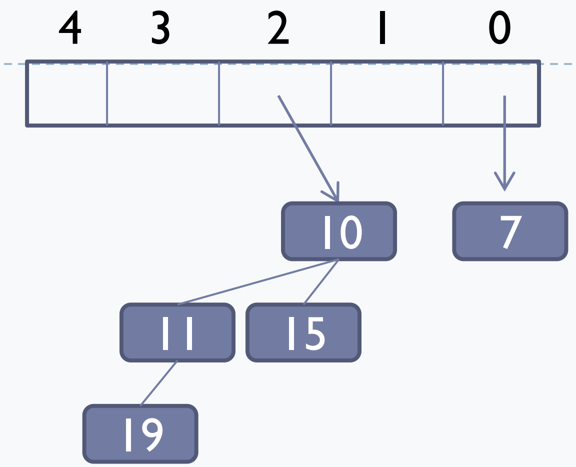
Delete the root node with the min element.
That gives us basically two binomial queues to merge

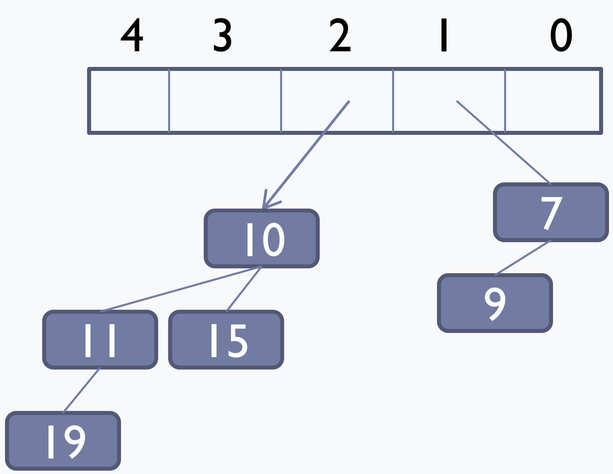
Binomial queue operations – delete minimum



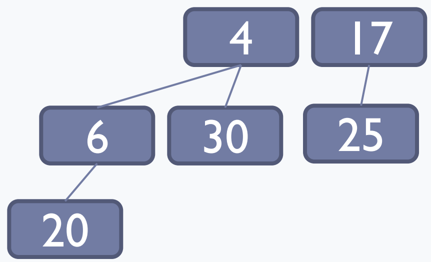
Binomial queue operations – delete minimum

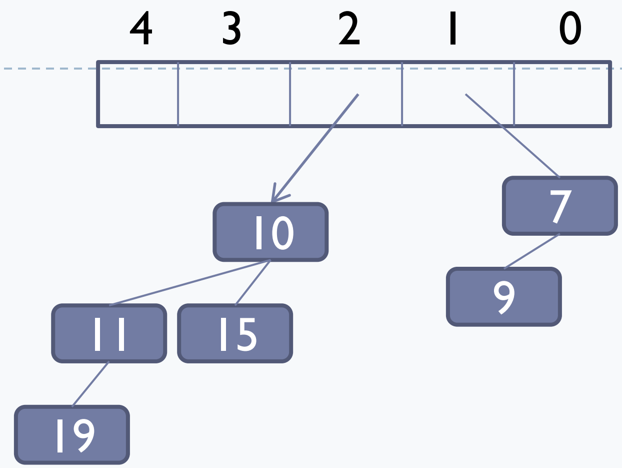


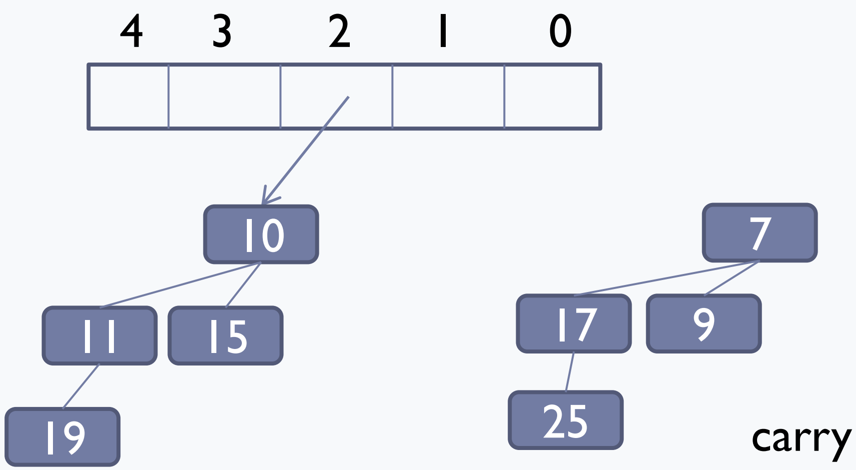




Binomial queue operations – delete minimum

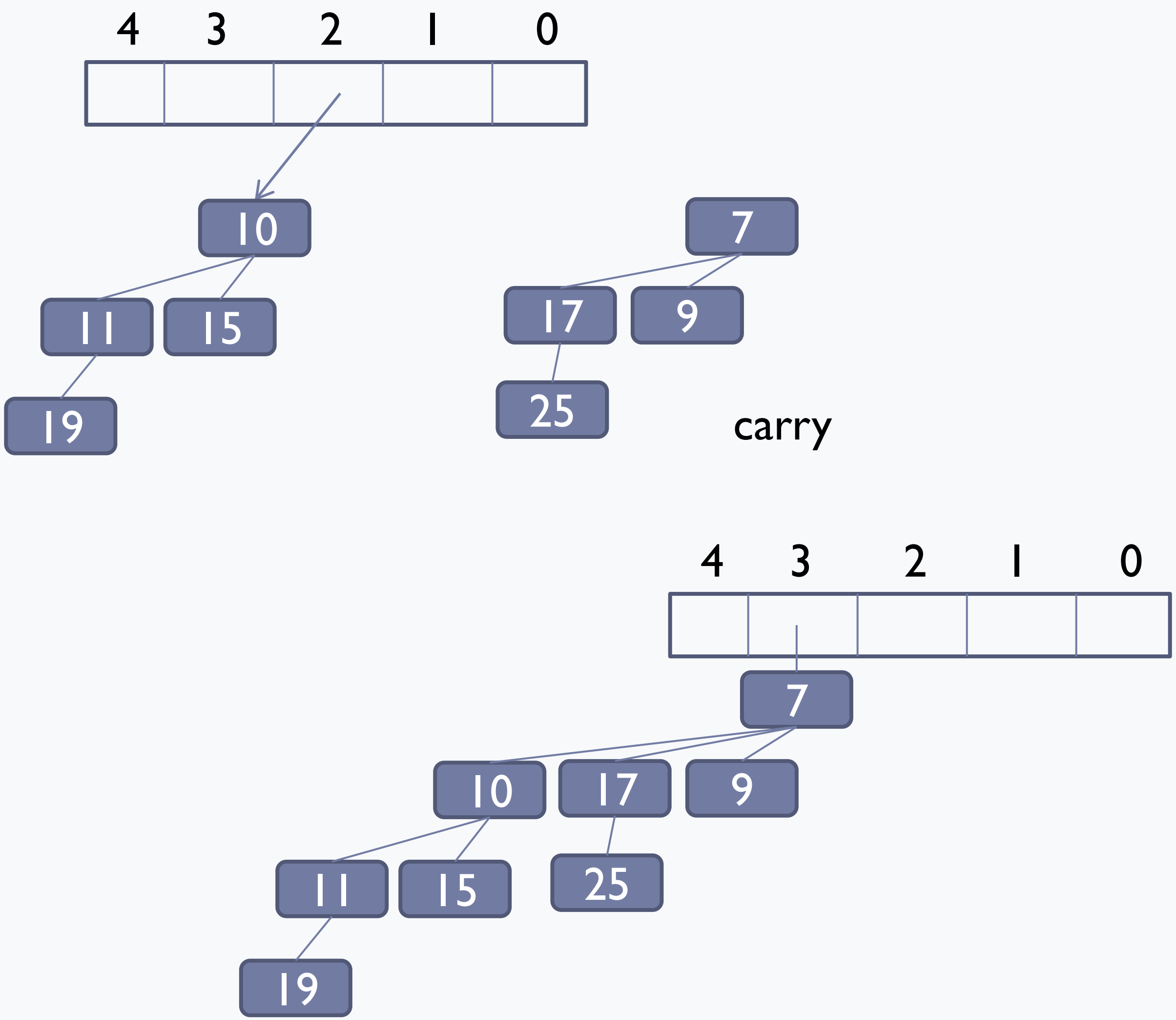




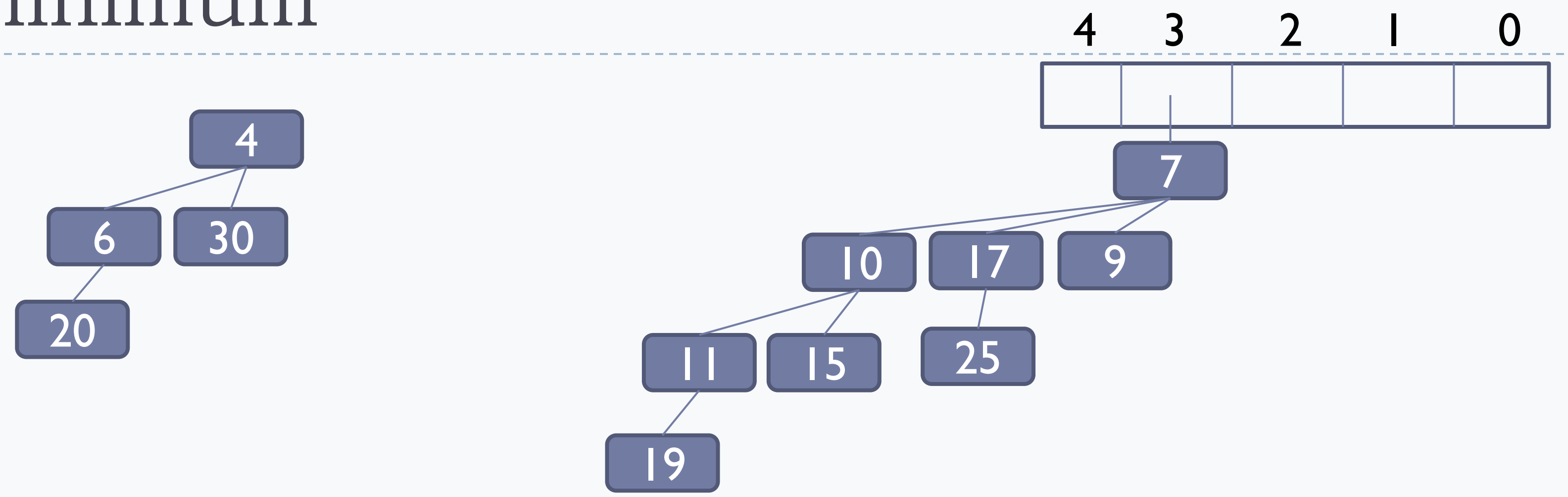


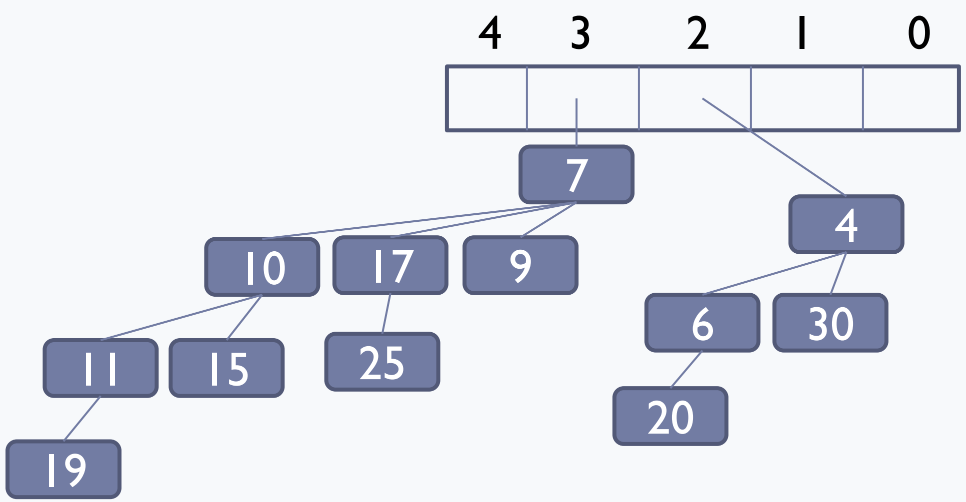
 Advanced material

Binomial queue operations – delete minimum



Binomial queue operations – delete minimum





 Advanced material

Binomial queues: delete minimum

- Worst case complexity
- Find the smallest of the $O(\log n)$ roots
- Remove the tree from the binomial queue
- Then delete the root node, which will give you two binomial trees
- merge the resulting two binomial trees into the remaining binomial queue
- Overall $O(\log n)$

Priority queues

operation	Unsorted doubly linked list	Red black tree	heap	Binomial queue	
insert	$O(1)$	$O(\log n)$	$O(\log n)$	$O(\log n)$	
Find min	$O(n)$	$O(\log n)$	$O(1)$	$O(\log n)$	
Delete min	$O(n)$	$O(\log n)$	$O(\log n)$	$O(\log n)$	
meld	$O(1)$	$O(n)$	$O(n)$	$O(\log n)$	