

Algoritmer og Datastrukturer 1

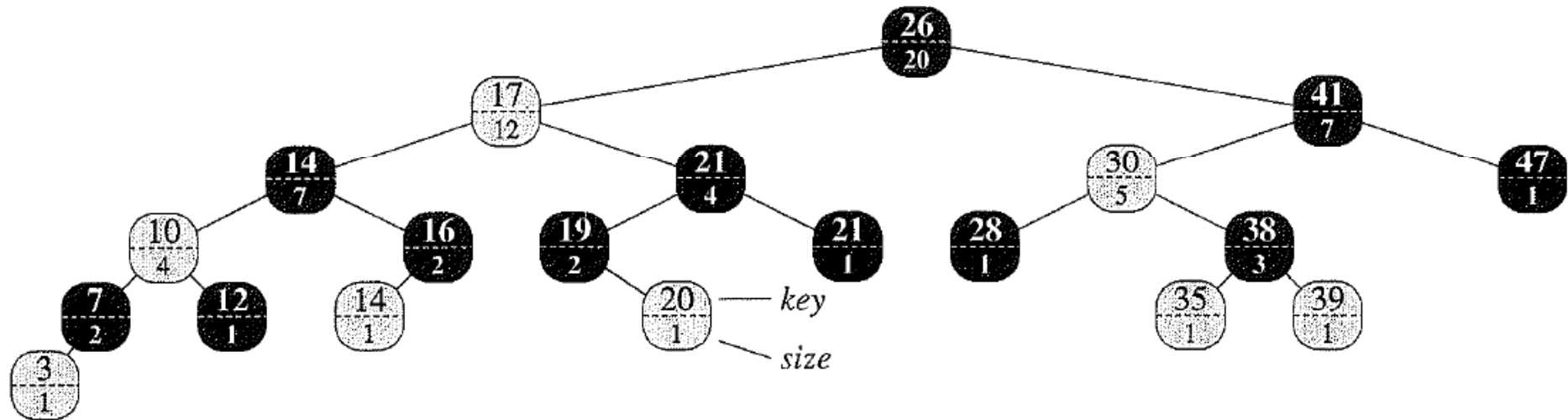
Gerth Stølting Brodal

Dynamisk Rang & Interval Træer [CLRS, kapitel 14]



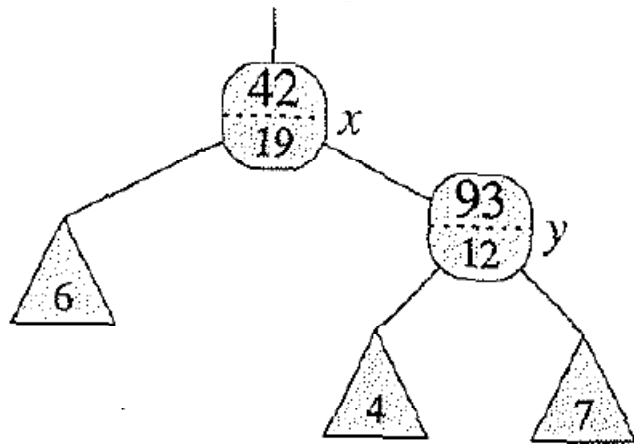
AARHUS UNIVERSITET

Dynamisk Rang



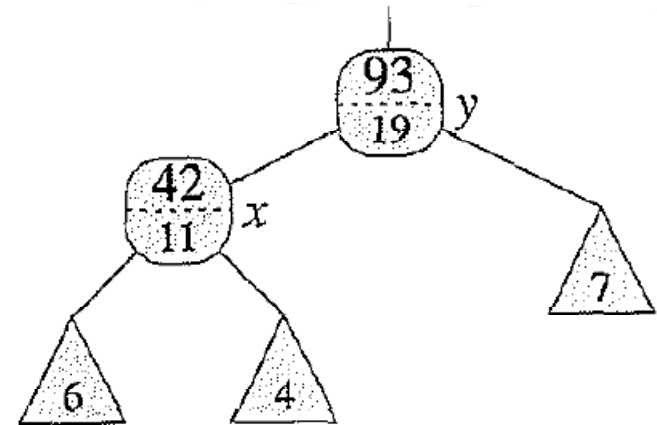
- Find det *i*'te mindste, indsættelser, slettelser
- Vedligehold i rød-sort søgetræ
- Udvid hver knude med *størrelse af undertræerne*

Dynamisk Rang



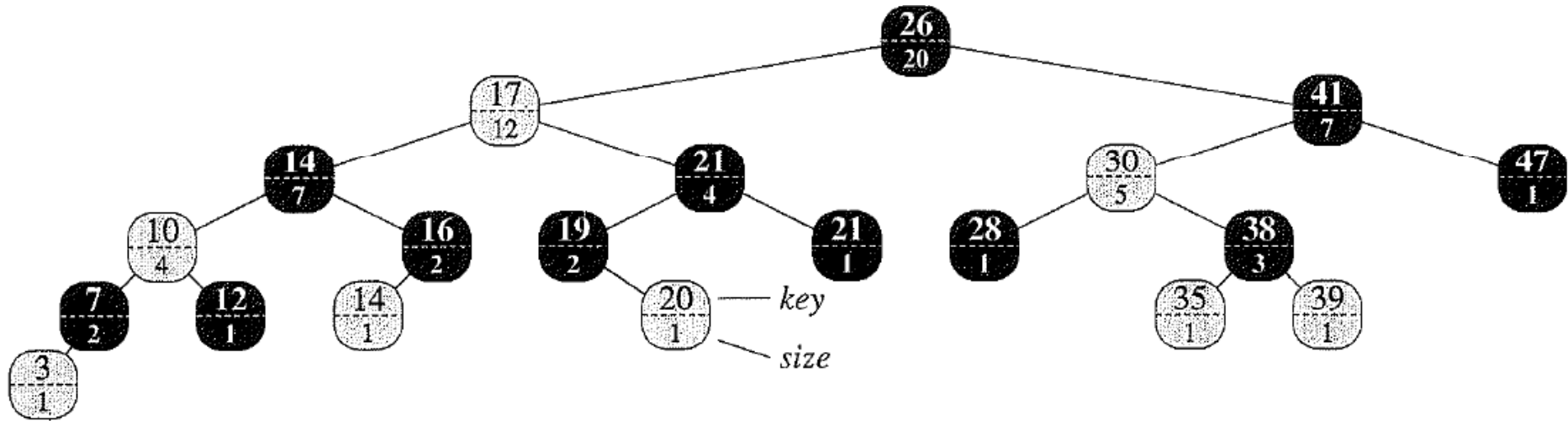
LEFT-ROTATE(T, x)

RIGHT-ROTATE(T, y)



- Indsættelse/slettelse: opdater **size** på stien til roden
- Under rebalancering af det rød-sortede træ, vedligehold **size** under **rotationer**

Dynamisk Rang



OS-RANK(T, x)

```

1   $r = x.left.size + 1$ 
2   $y = x$ 
3  while  $y \neq T.root$ 
4      if  $y == y.p.right$ 
5           $r = r + y.p.left.size + 1$ 
6       $y = y.p$ 
7  return  $r$ 

```

OS-SELECT(x, i)

```

1   $r = x.left.size + 1$ 
2  if  $i == r$ 
3      return  $x$ 
4  elseif  $i < r$ 
5      return OS-SELECT( $x.left, i$ )
6  else return OS-SELECT( $x.right, i - r$ )

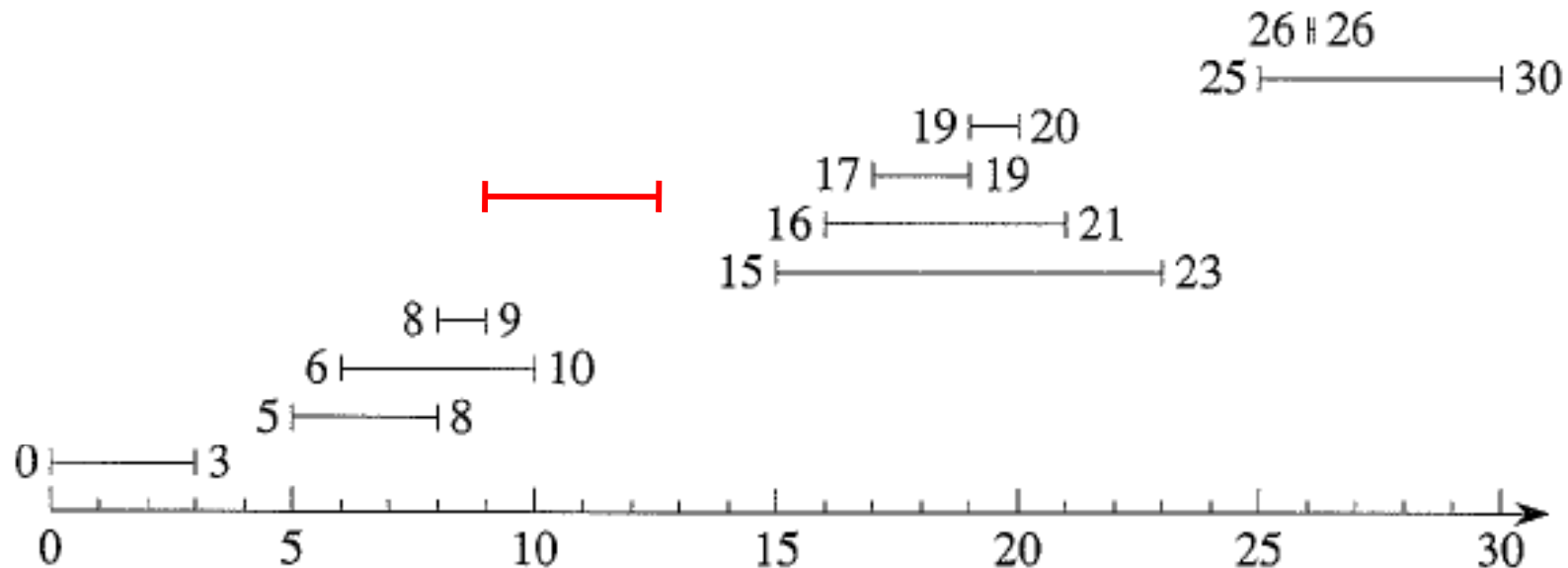
```

Dynamisk Rang

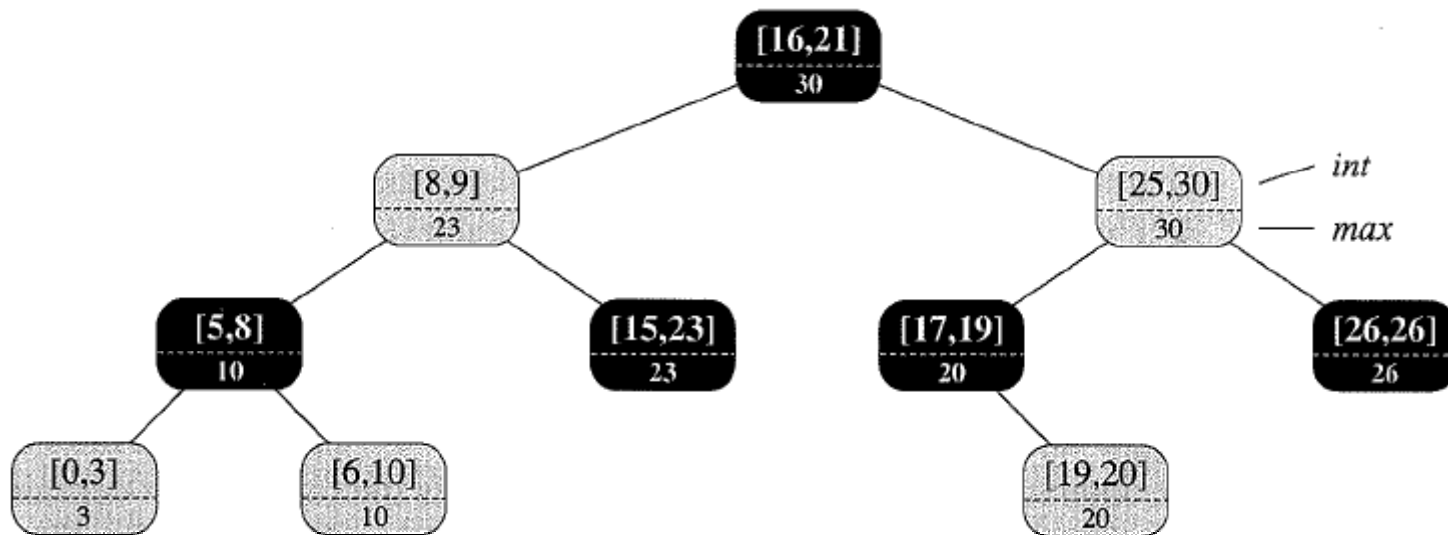
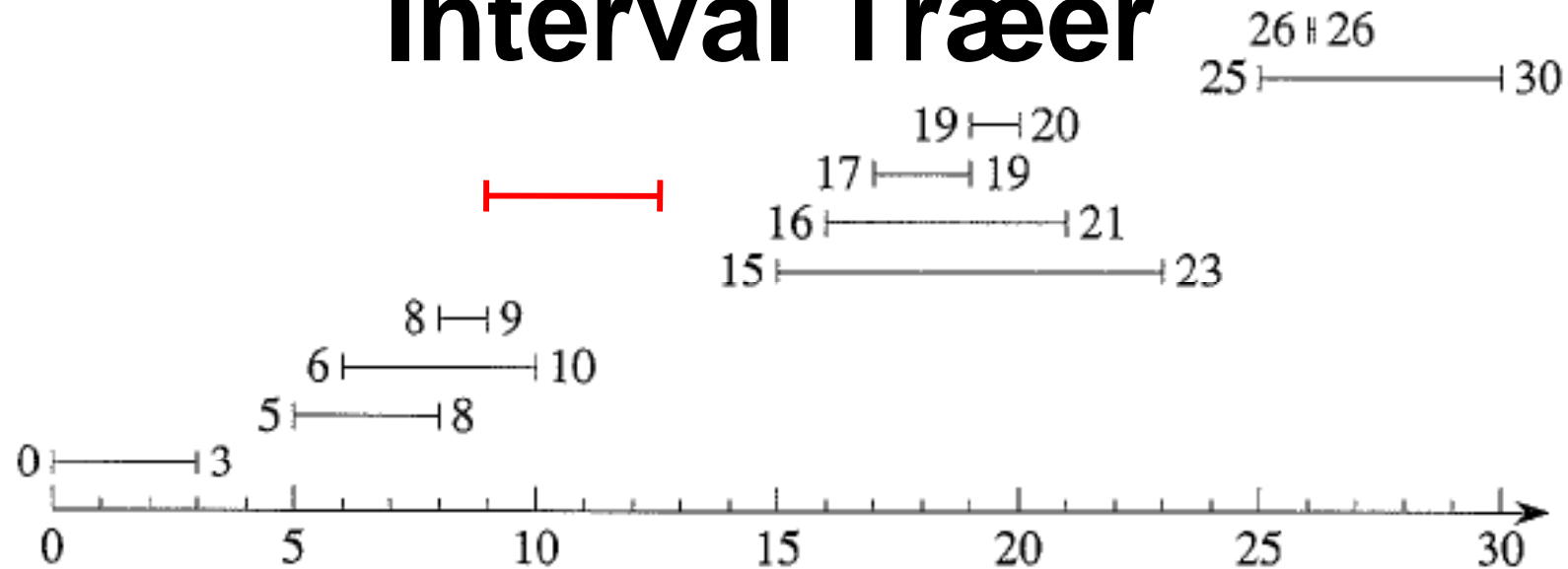
Select(S, i)	$O(\log n)$
Rank(S, x)	
Insert(S, x)	
Delete(S, x)	

Interval Træer

- Vedligehold en mængde af intervaller
- Indsæt og slet indsatte intervaller
- Find et interval der overlapper med et givet **interval**



Interval Træer

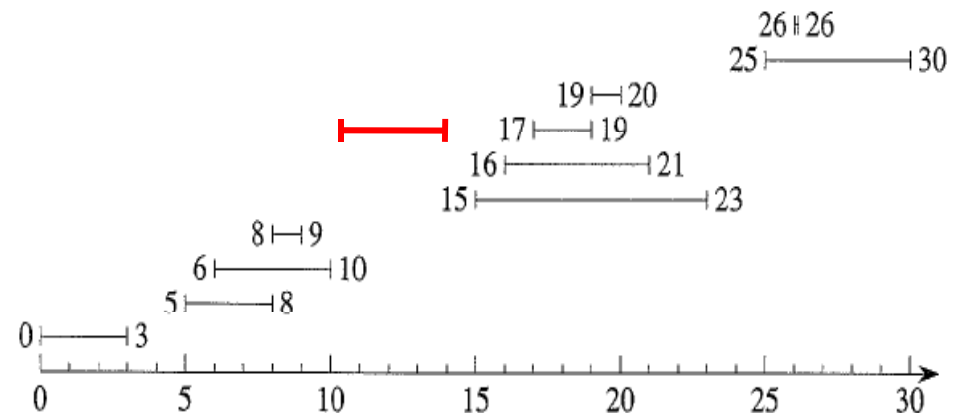
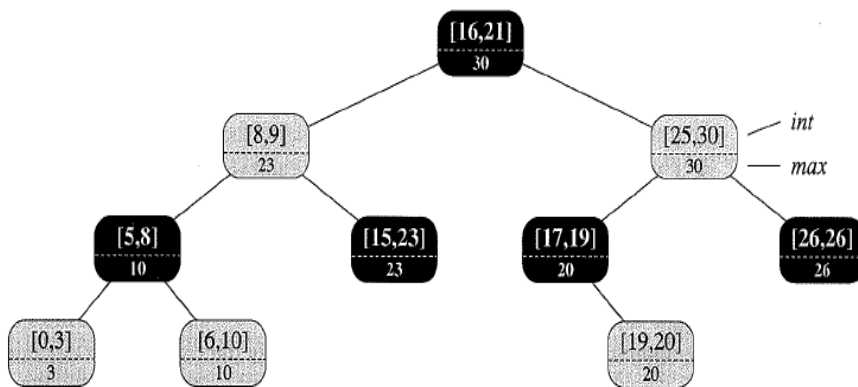


- Søgetræ over intervallernes **venstre endepunkt**
- Hver knude gemmer yderligere **maximum højre endepunkt** for et interval i undertræet

Interval Træer

INTERVAL-SEARCH(T, i)

- 1 $x = T.root$
- 2 **while** $x \neq T.nil$ and i does not overlap $x.int$
- 3 **if** $x.left \neq T.nil$ and $x.left.max \geq i.low$
- 4 $x = x.left$
- 5 **else** $x = x.right$
- 6 **return** x



Interval Træer

Search(T, i)	$O(\log n)$
Insert(T, i)	
Delete(T, i)	