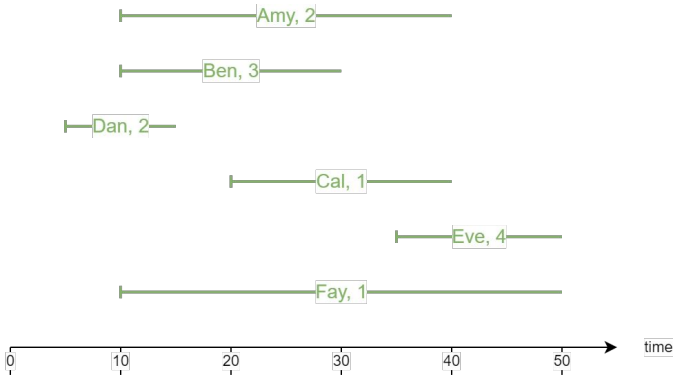
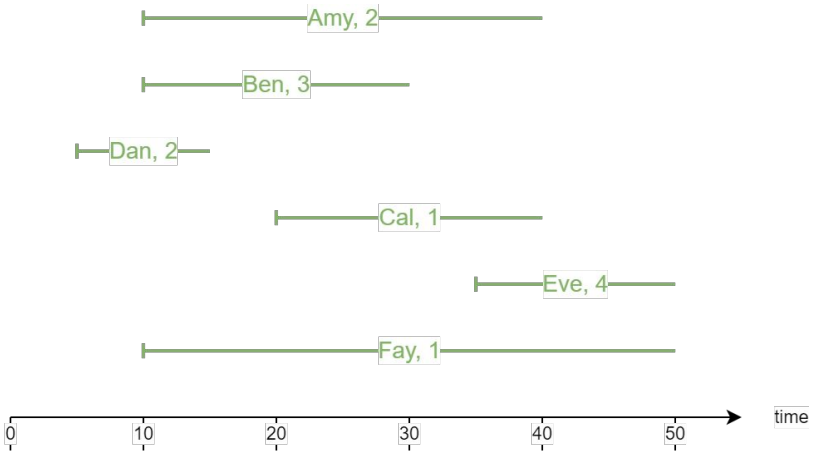
SB-Tree



Incremental computation and maintenance of temporal aggregates

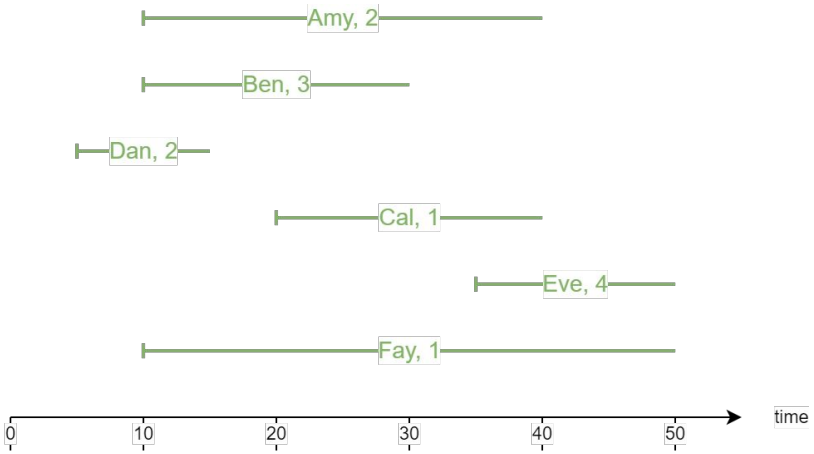
Mirko Bristle, Markus Eggimann

Context I: Temporal data



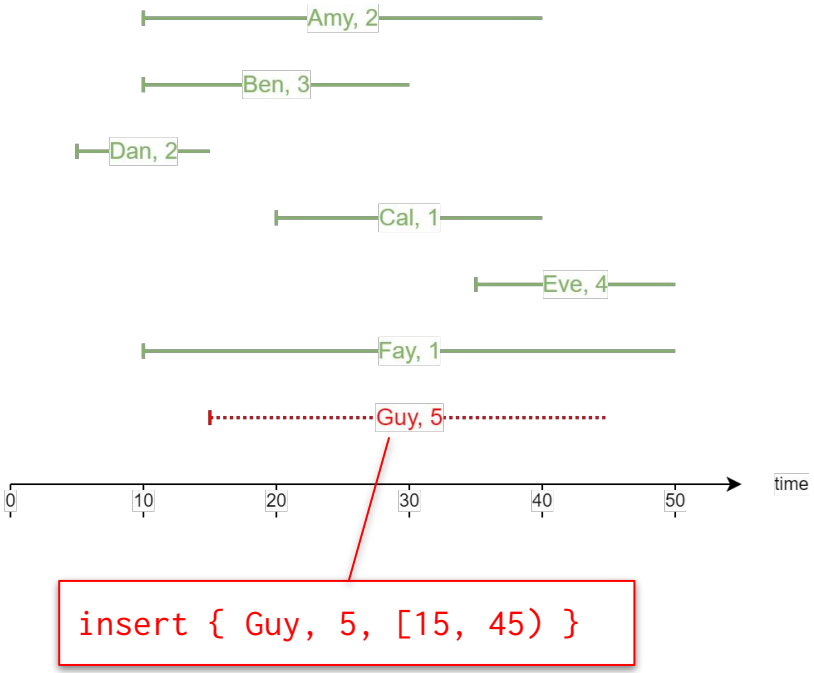
Patient	Dosage	Interval
Amy	2	[10, 40)
Ben	3	[10, 30)
Cal	1	[20, 40)
Dan	2	[5, 15)
Eve	4	[35, 45)
Fay	1	[10, 50)

Context II: Temporal aggregates



Sum dosage	Interval
0	[0, 5)
2	[5, 10)
8	[10, 15)
6	[15, 20)
7	[20, 30)
4	[30, 35)
8	[35, 40)
5	[40, 45)
1	[45, 50)
0	[50, …)

The problem



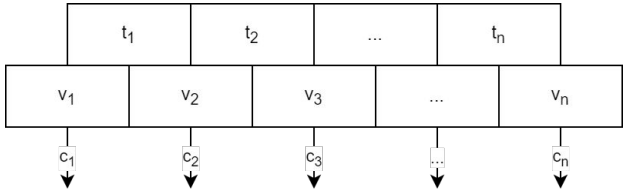
Sum dosage	Interval
0	[0, 5)
2	[5, 10)
8	[10, 15)
11	[15, 20)
12	[20, 30)
9	[30, 35)
13	[35, 40)
10	[40, 45)
1	[45, 50)
0	[50, …)

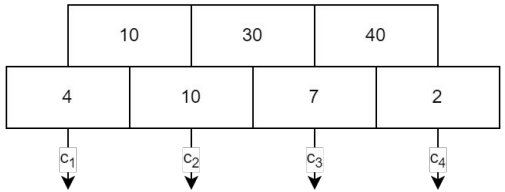
A solution

- Use a tree-based index for each aggregate
- Use a combination of B-trees and segment trees
- Allow fast lookups and efficient updates

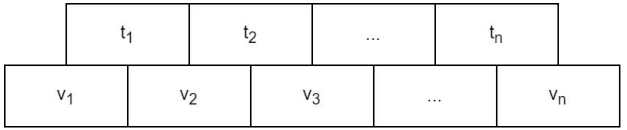
➔ Introducing the **SB-tree**!

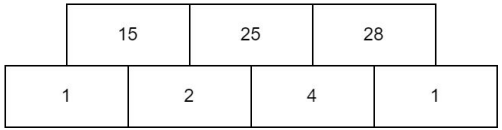
SB-tree: Building blocks





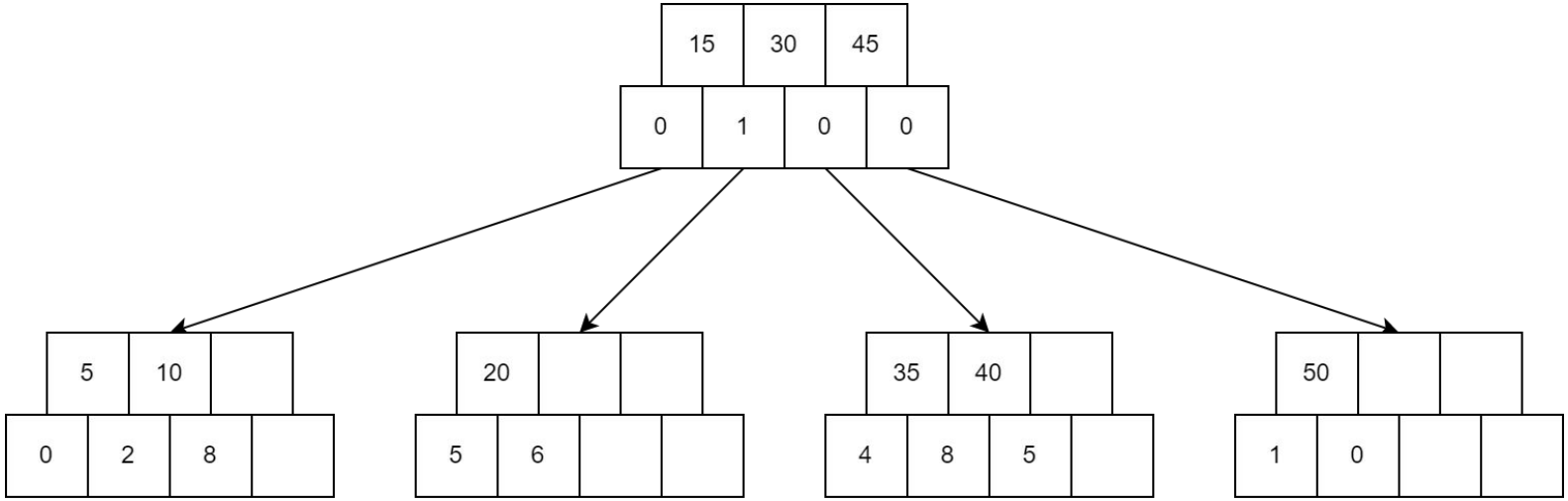
Interior node





Leaf node

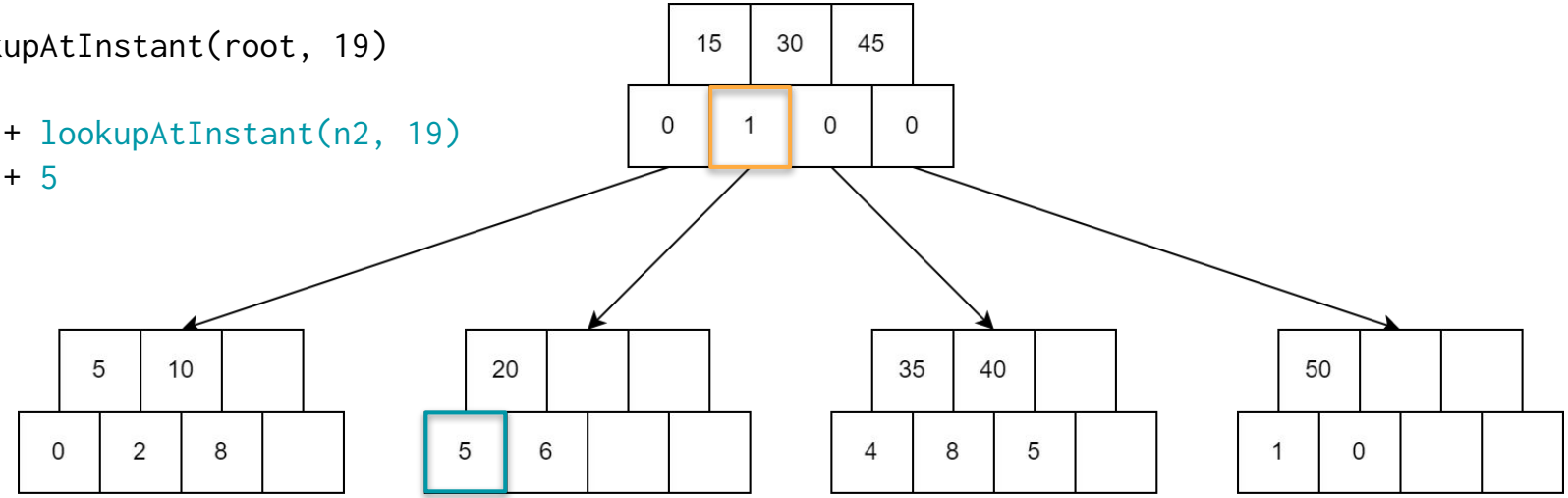
SB-tree: The complete structure



SB-tree: Lookup

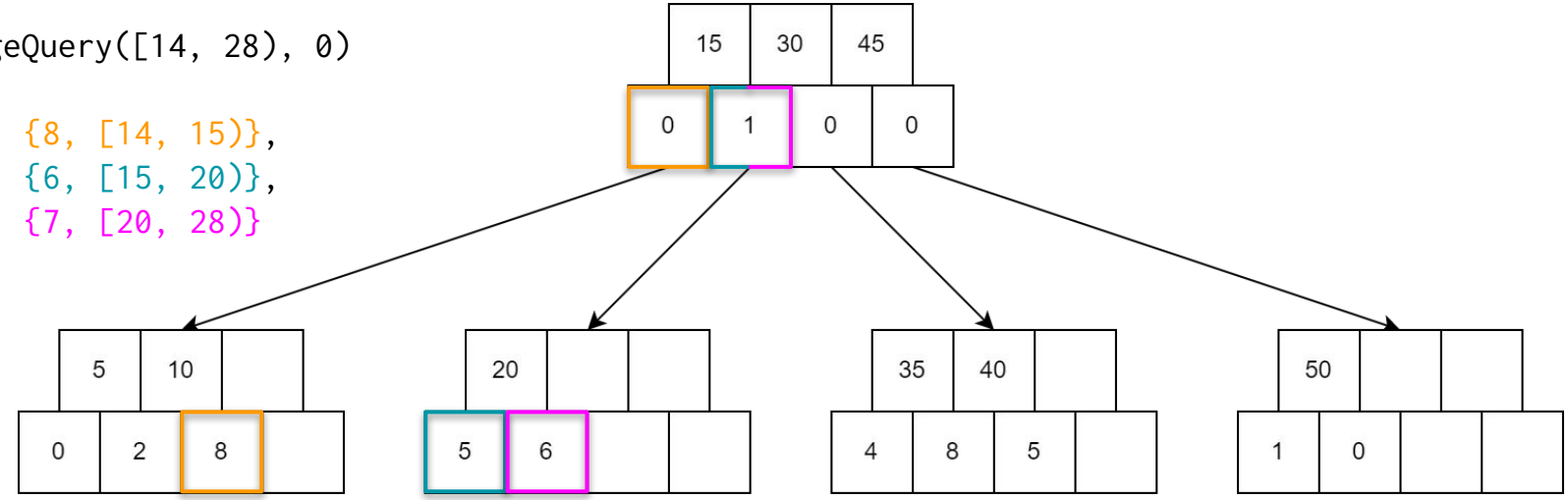
```
lookupAtInstant(root, 19)

= 1 + lookupAtInstant(n2, 19)
= 1 + 5
= 6
```



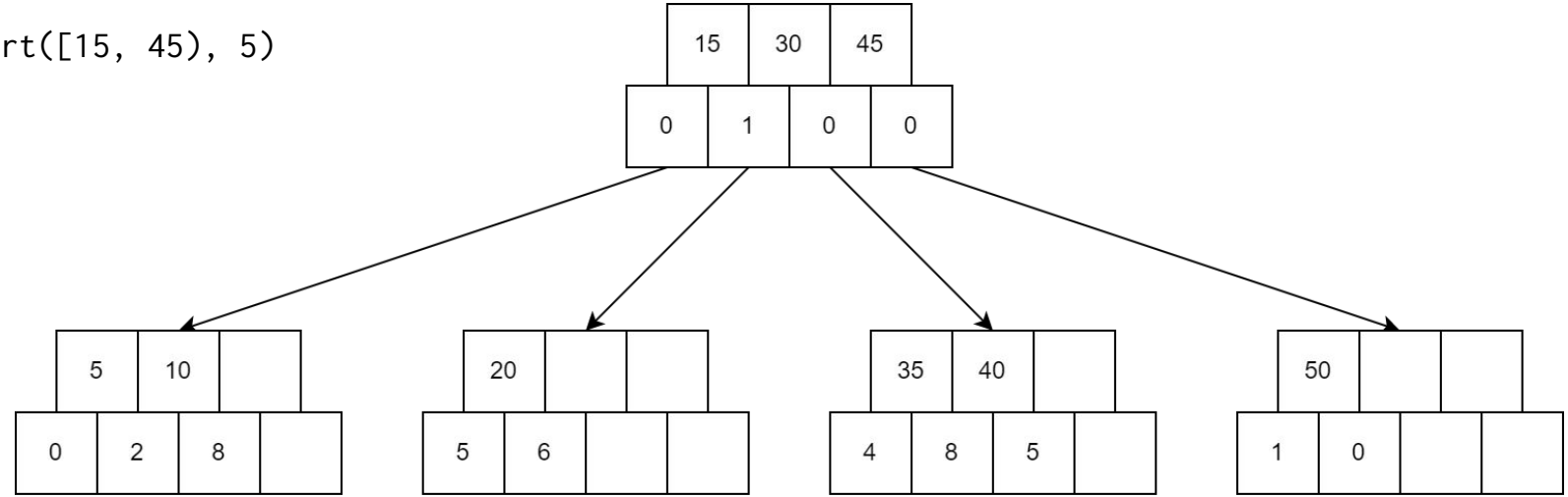
SB-tree: Range query

```
rangeQuery([14, 28), 0)
= [
      {8, [14, 15)},
      {6, [15, 20)},
      {7, [20, 28)}
  ]
```



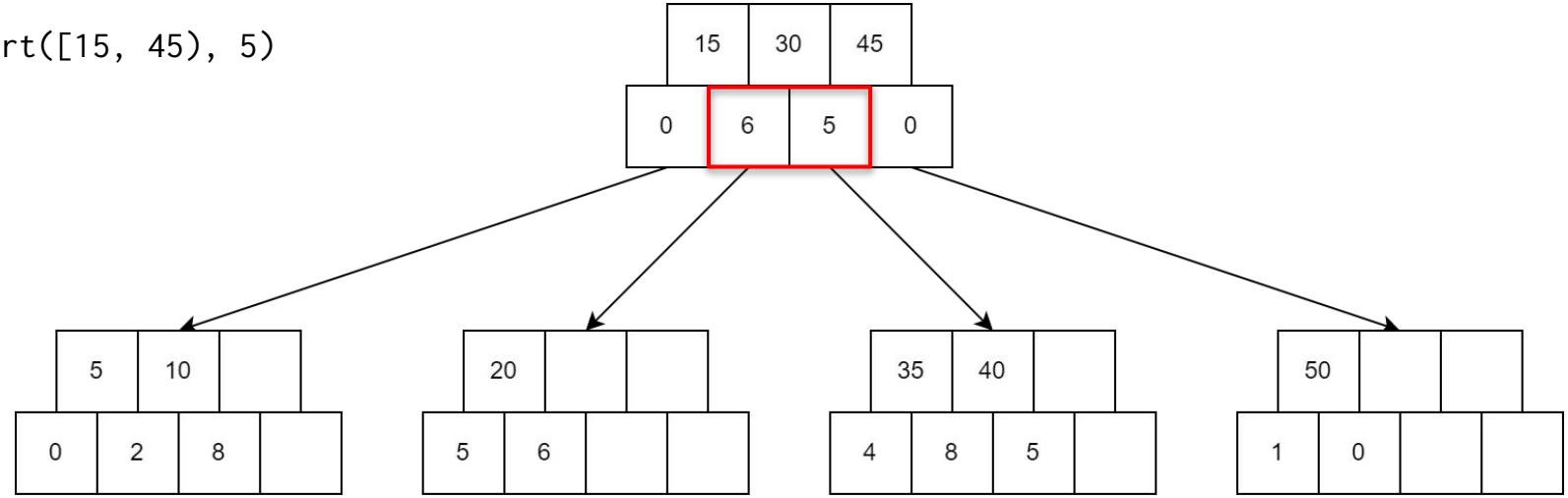
SB-tree: Insert I

```
insert([15, 45), 5)
```



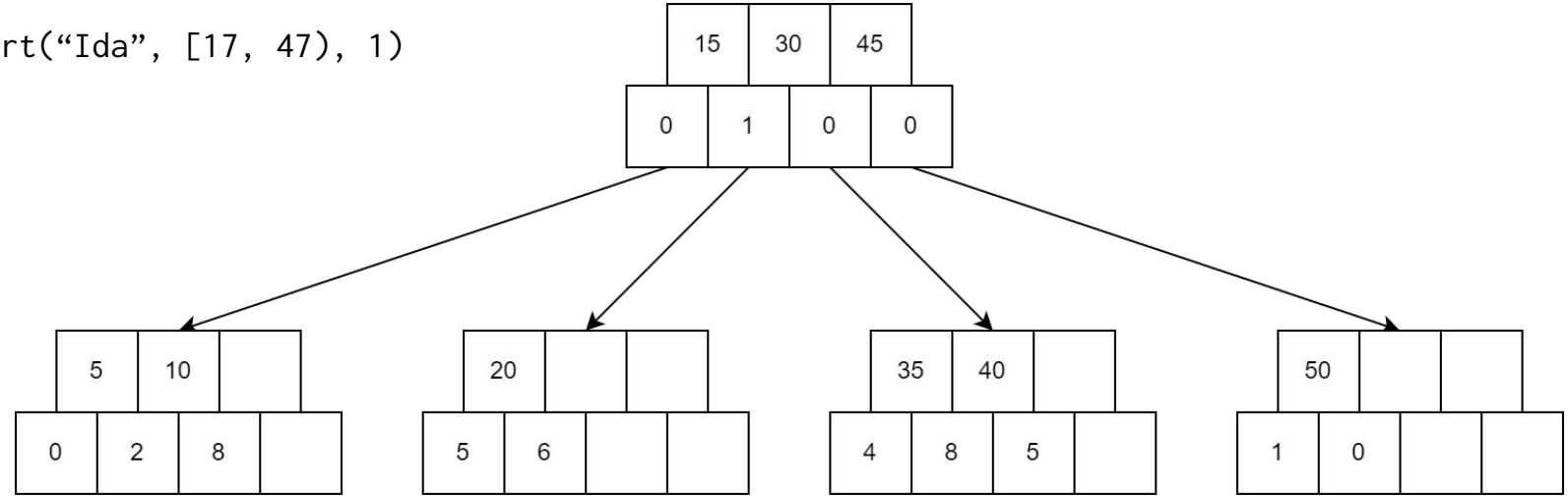
SB-tree: Insert I

```
insert([15, 45), 5)
```



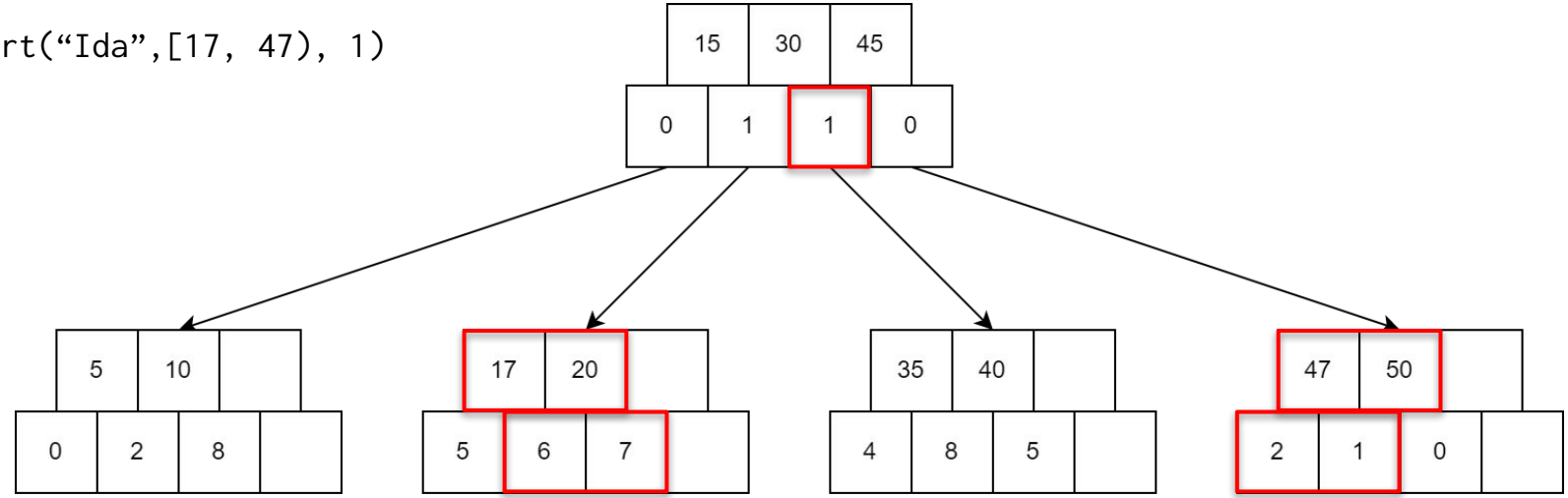
SB-tree: Insert II

```
insert(“Ida”, [17, 47), 1)
```



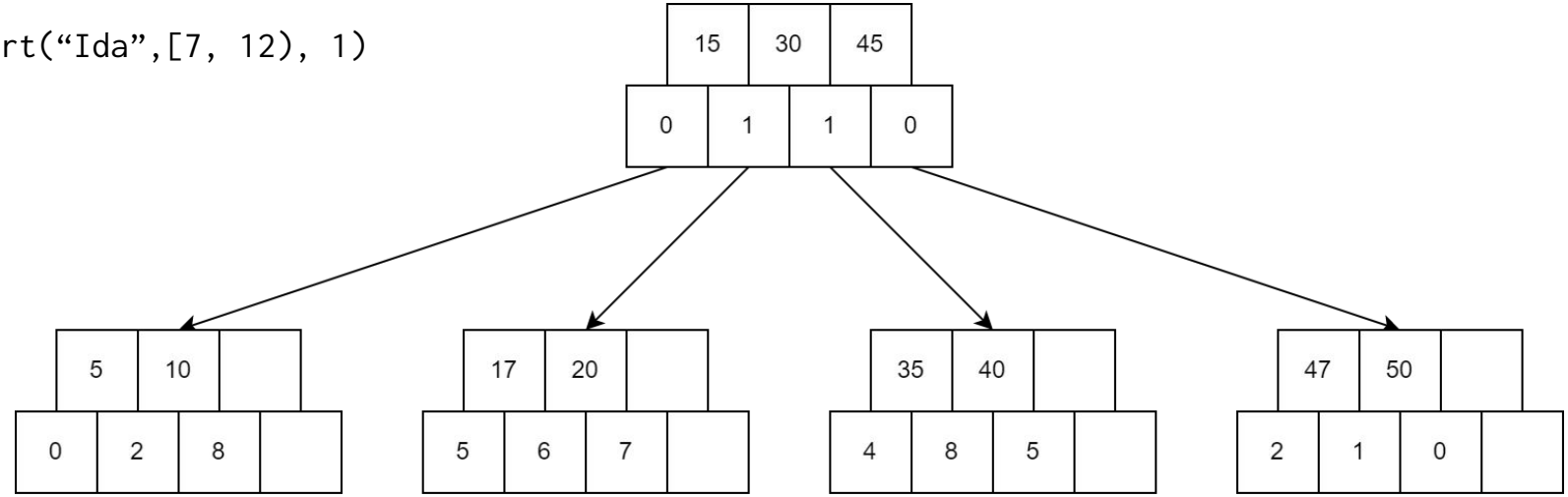
SB-tree: Insert II

```
insert("Ida",[17, 47), 1)
```



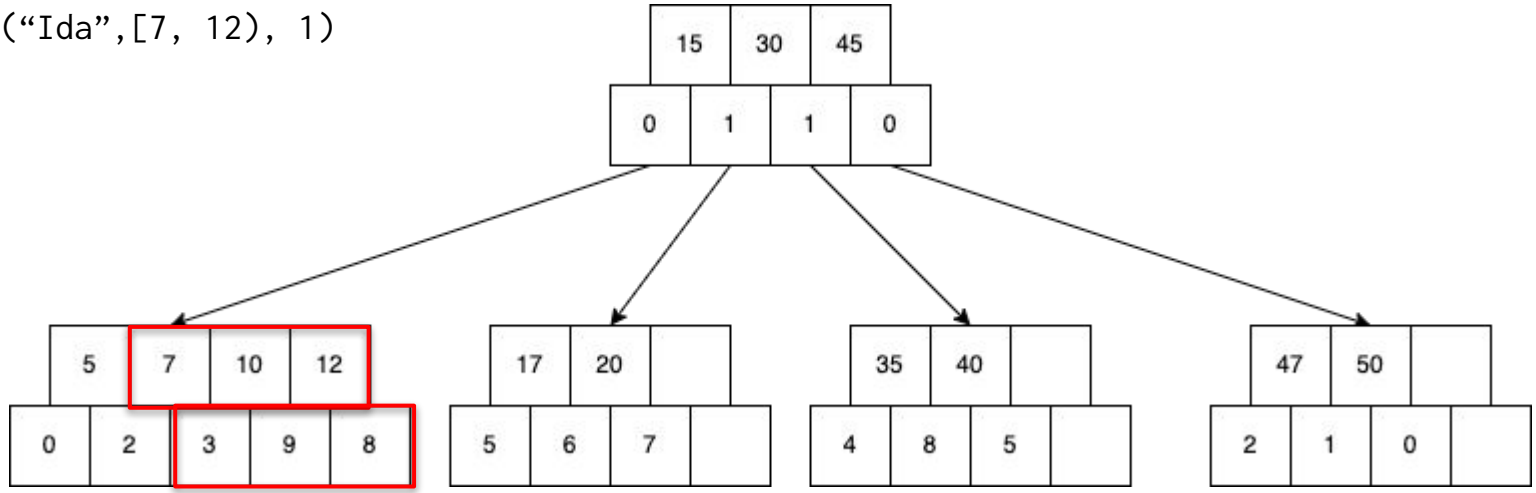
SB-tree: Node splits

```
insert(“Ida”,[7, 12), 1)
```



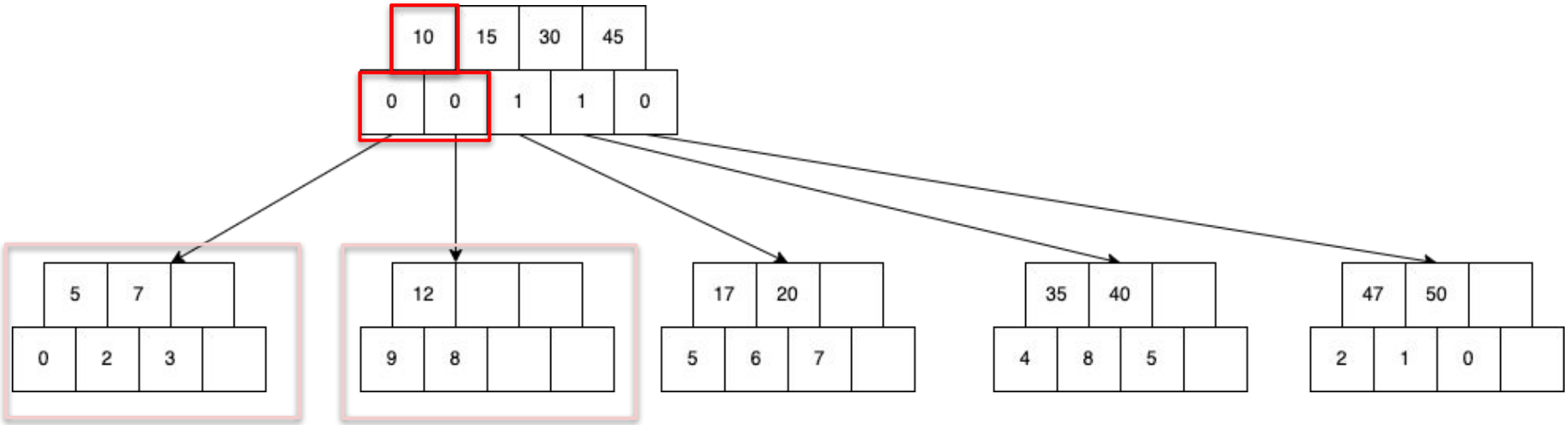
SB-tree: Node splits

```
insert(“Ida”,[7, 12), 1)
```



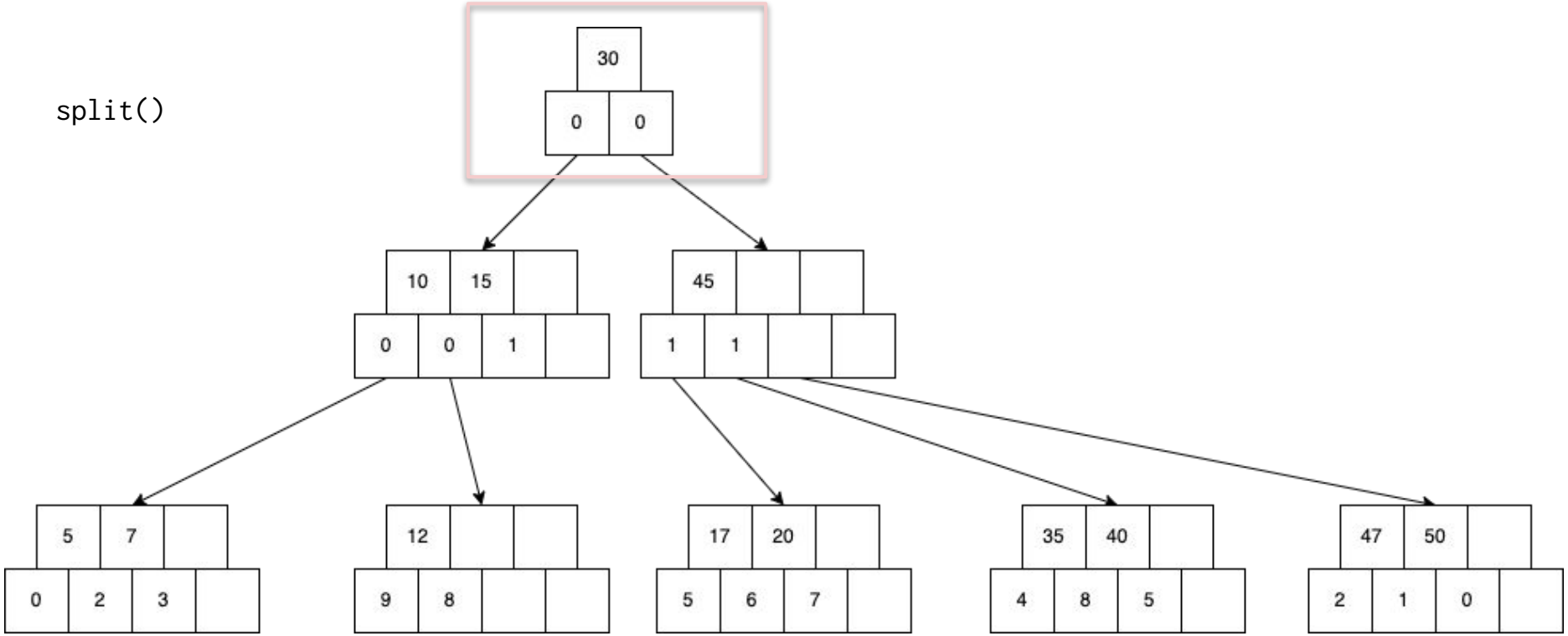
SB-tree: Node splits

`split()`



SB-tree: Node splits

`split()`

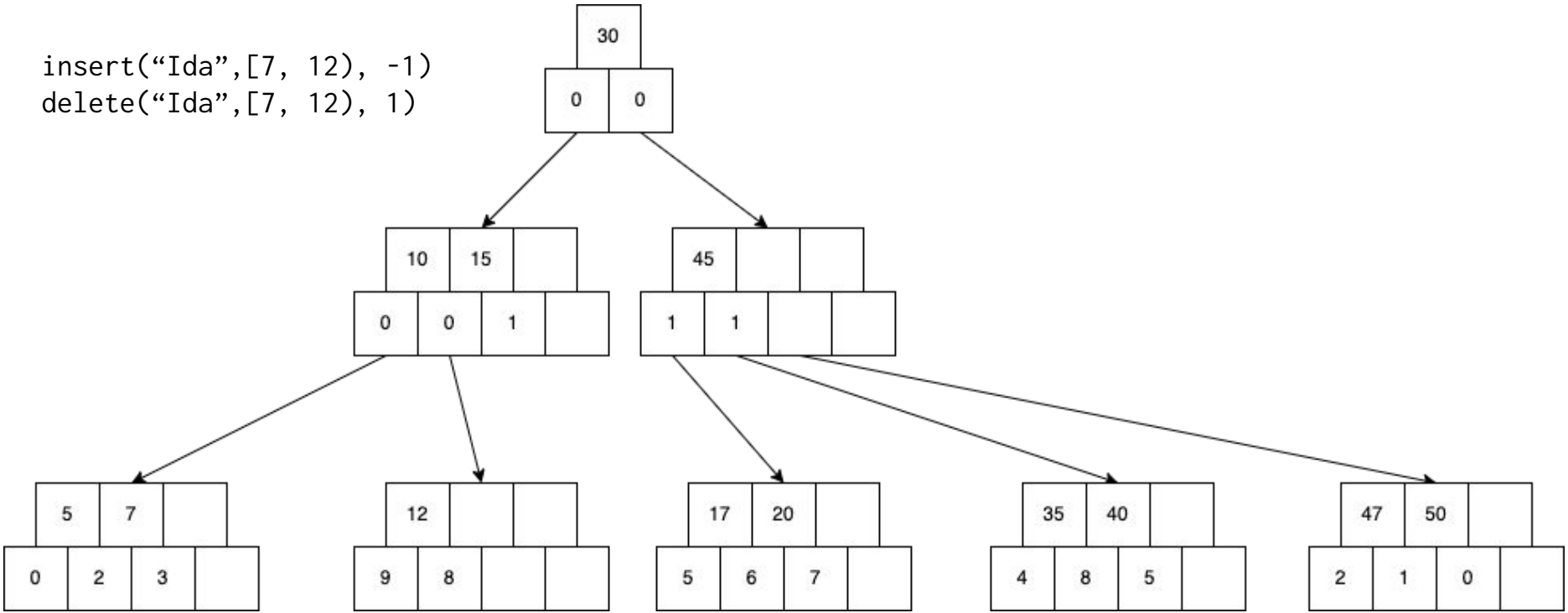


SB-tree: Delete

- Like inserts!
- Example:
  - delete { “lda”, 1, [17, 47) }
  - is equivalent to insert { . , -1, [17, 47) }
- But underflows may occur!

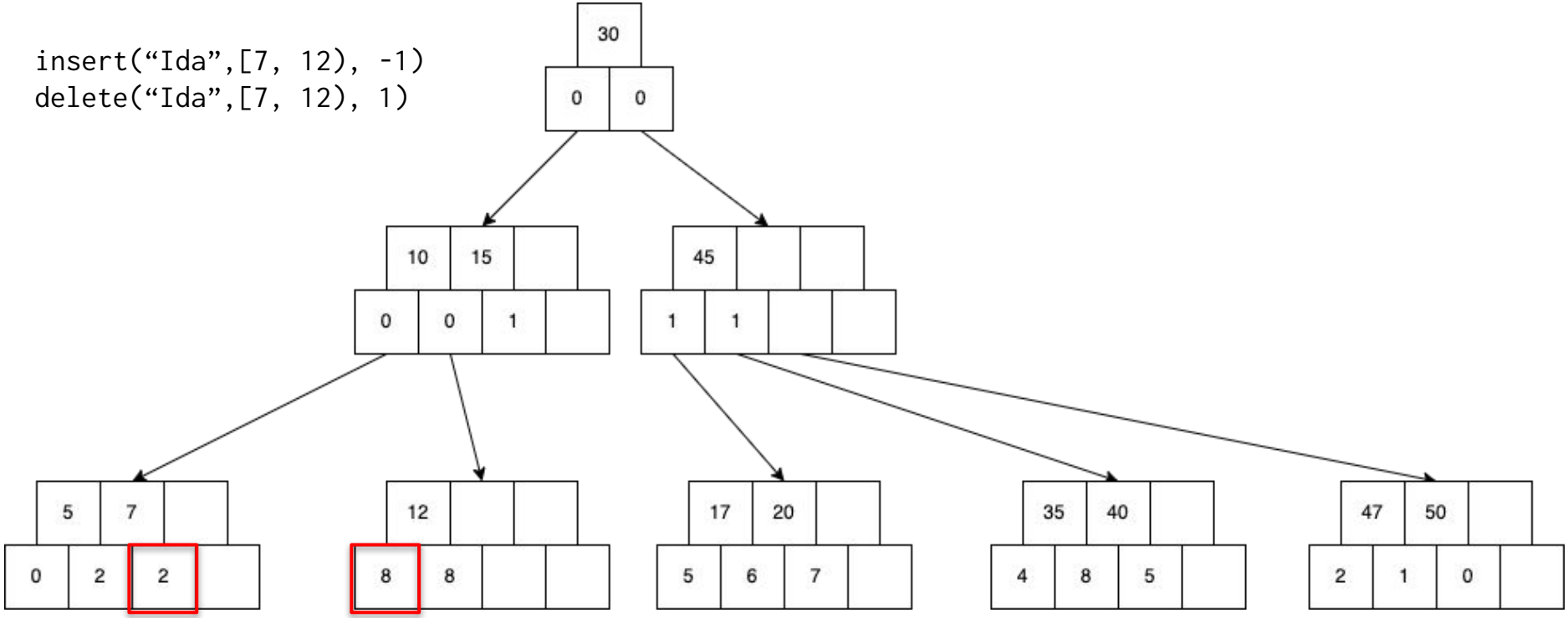
SB-tree: Node merges

```
insert(“Ida”,[7, 12), -1)
delete(“Ida”,[7, 12), 1)
```



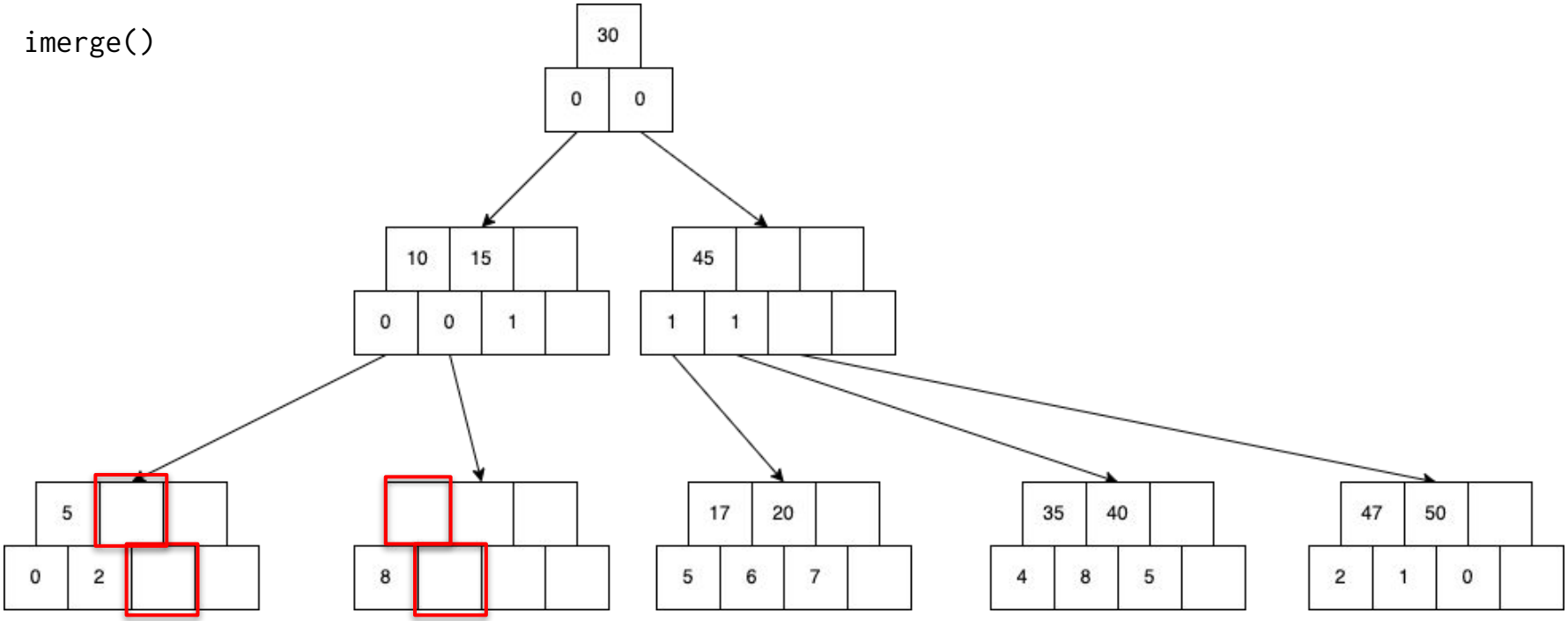
SB-tree: Node merges

```
insert(“Ida”,[7, 12), -1)
delete(“Ida”,[7, 12), 1)
```



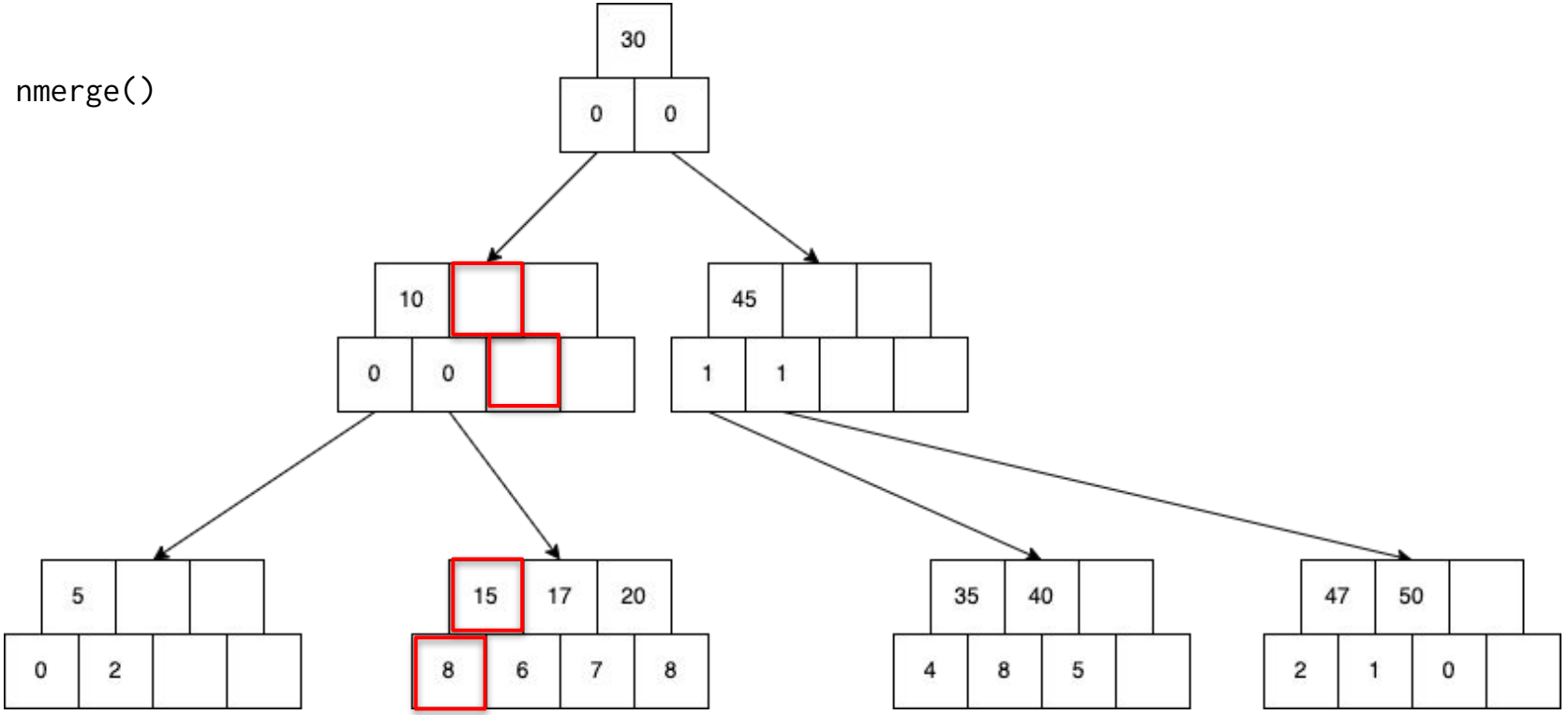
SB-tree: Node merges

`imerge()`



SB-tree: Node merges

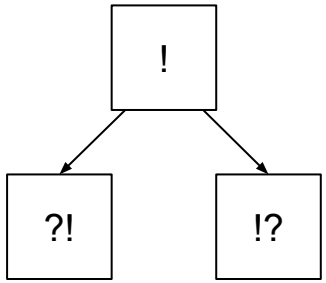
`nmerge()`



Performance

- Lookup: O(h)
- Range query: O(h + r)
- Insert: O(h)
- Delete: O(h)

h: height of the tree
r: number of leafs intersecting with the query range



Discussion

References

Incremental computation and maintenance of temporal aggregates
Jun Yang, Jennifer Widom
The VLDB Journal (2003) 12, p262-283