Persistent Stack

Main Concept in Persistent Stack

Kushal Das 30/12/2021

Here I have used pointer machine to implement my fully persistent Stack

I have taken some basic functions of stack

- Push
- Pop
- IsEmpty
- Print

I have taken the print function just to see what the stack was in a particular version.

In this I have implemented a list using pointer machine in which each node has:

- Data
- Next pointer
- Version
- Back pointer
- Modifications log
    - Data
    - Next and Back pointers
    - Version

Then I implemented an insert at an index and delete at an index in any version functions.

Then for stack the index is 0 i.e. insert at start and delete at start.

Now lets come to Javascript

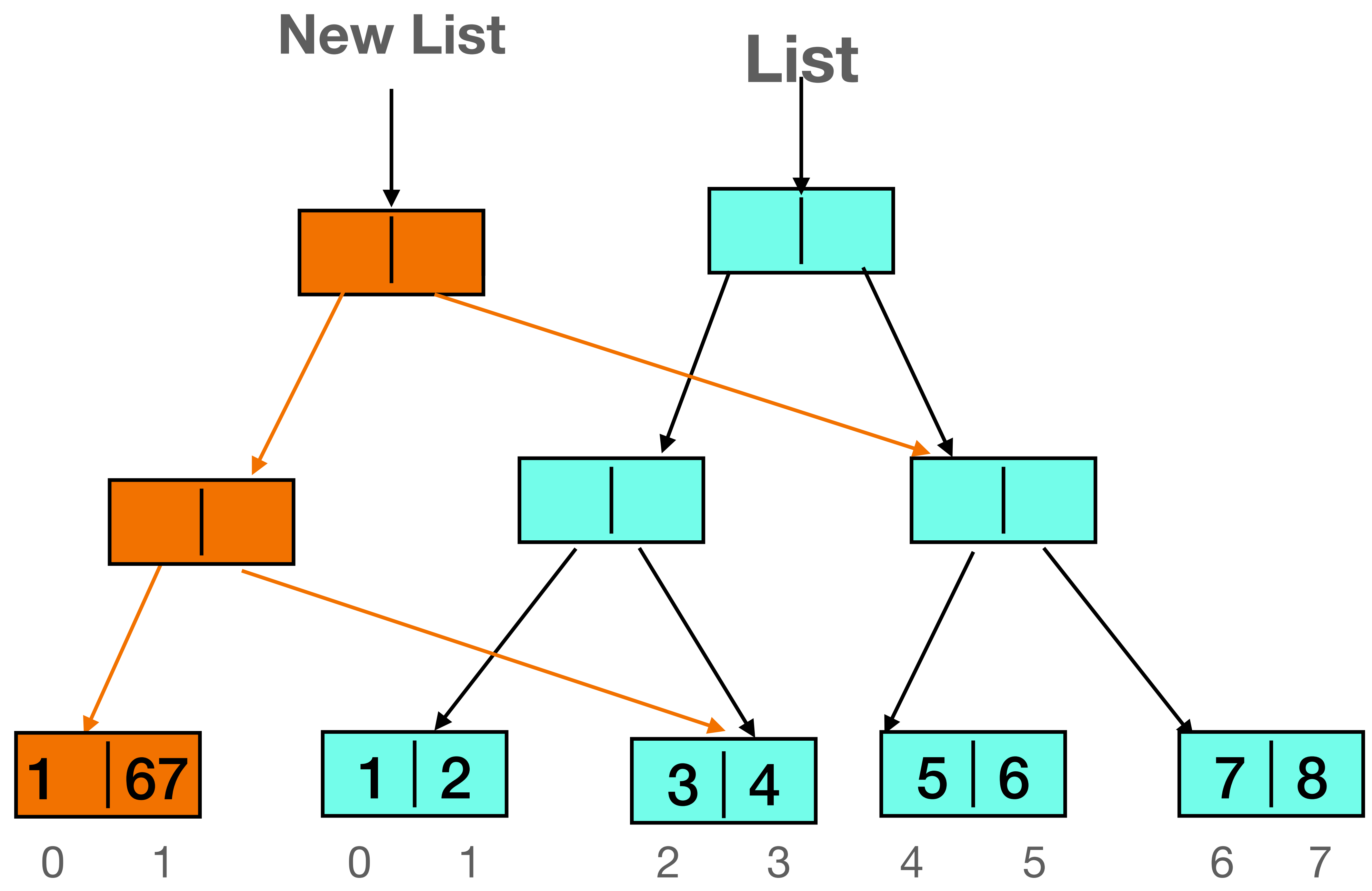
Suppose we want to make a list of some numbers

list=[1,2,3,4,5,6,7,8]

Now suppose we want to change the number at index 1 i.e. 2 to 67.

Ok so now lets break our list of numbers in size of 2 → 1,2 3,4 5,6 7,8

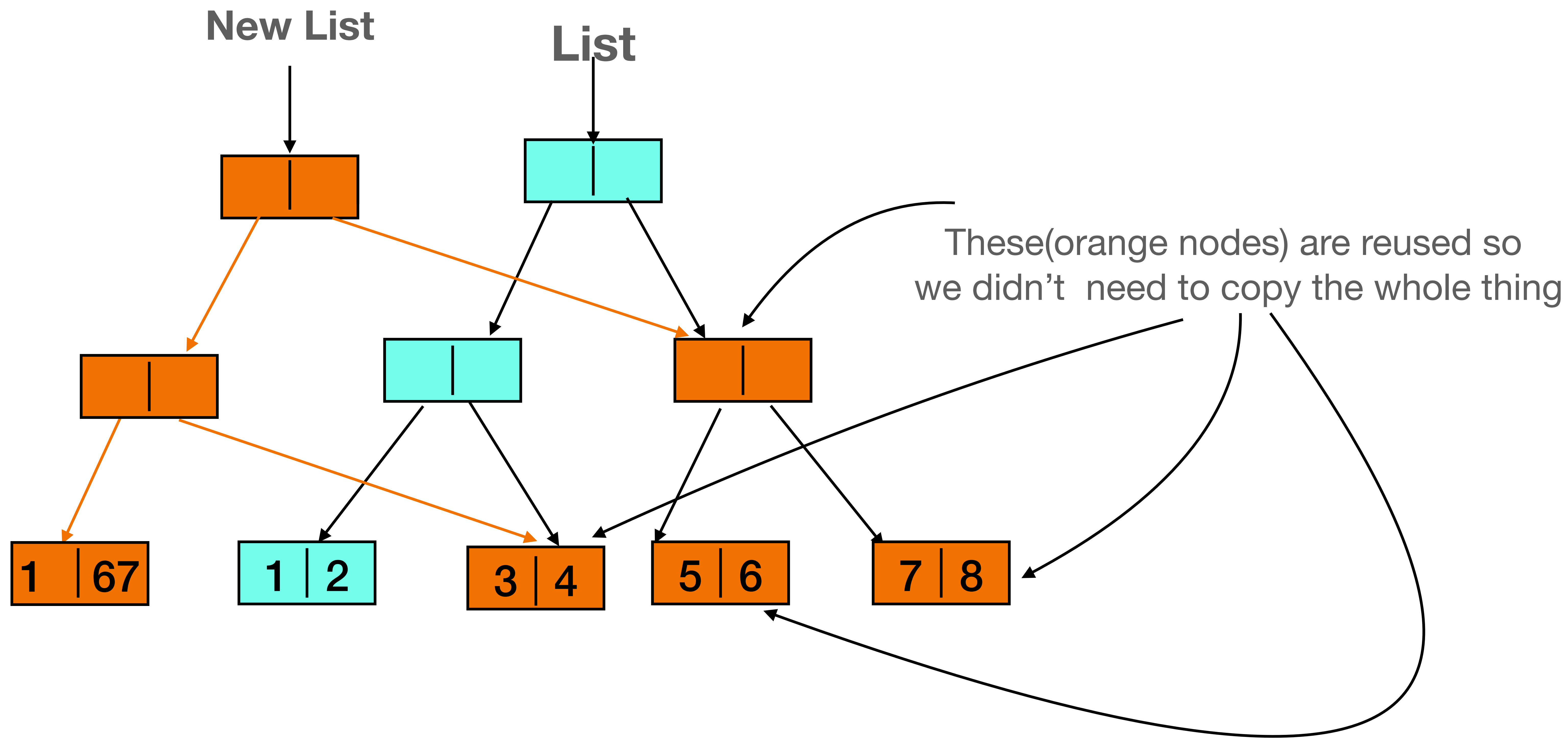
So now we will update the value 2 to 67



1st the node is copied

Then path copying happens

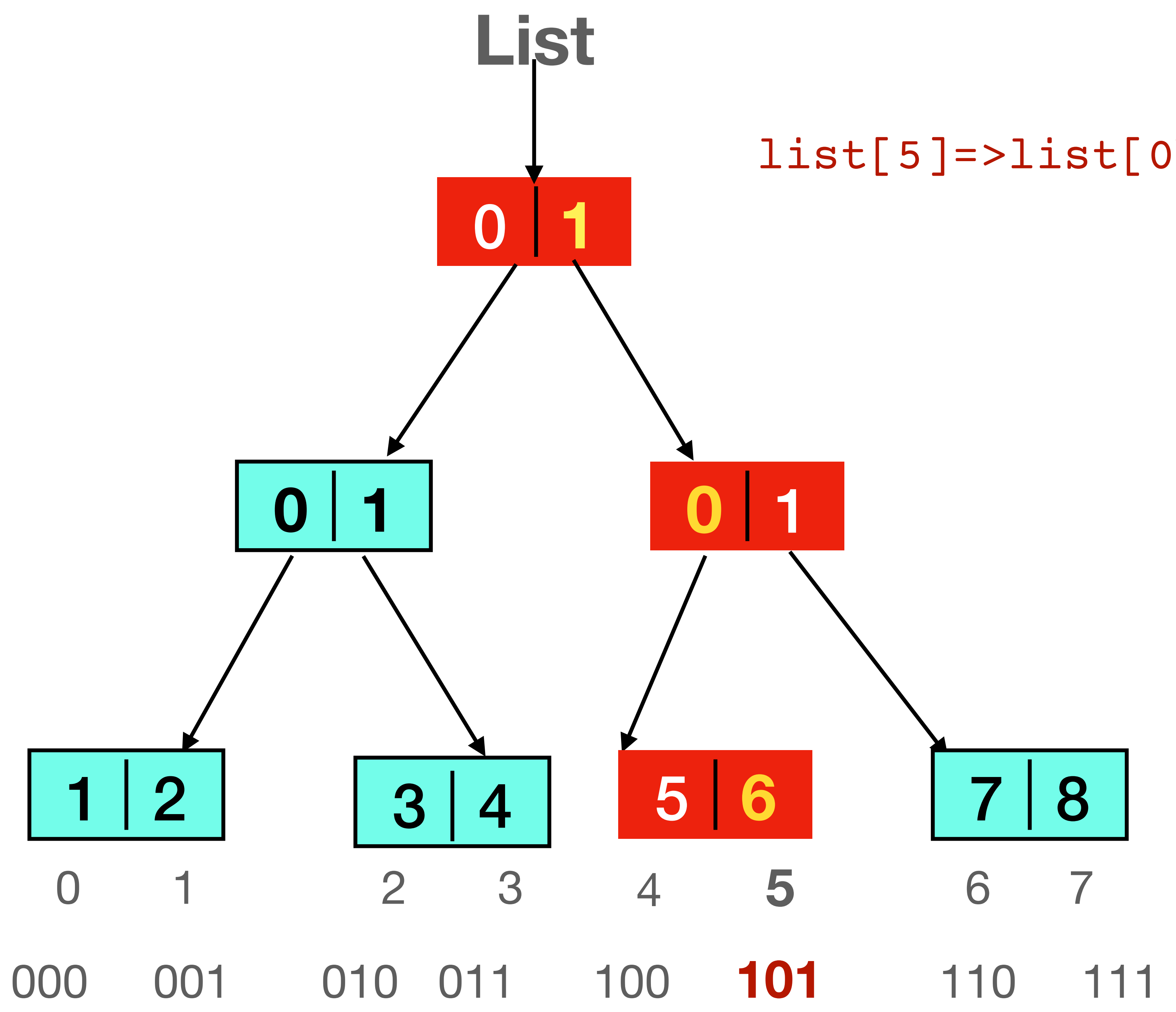
So now the list is



These(orange nodes) are reused so we didn’t need to copy the whole thing

So now if we want to get a value at any index say 5
We can do that easily using hashing 5 to its binary value 0b101

We will go from 1 ⟶ 0 ⟶ 1 to get
our value I.e. 5



```
list[5]=>list[0b101]=>list->1->0->1=>6
```

In this way we can reach to any node quickly and return the value in a less time using hashing