CS3A Sem Test II

Date: 2nd May 2024

Time : 14h00 – 16h00

Topics:

- Trees and Binary Trees
- Priority Queues, Heaps and Adaptable Priority Queues
- Maps and Hash Tables
- Dictionary and Skip Lists
- Search Trees(Binary Search Trees)

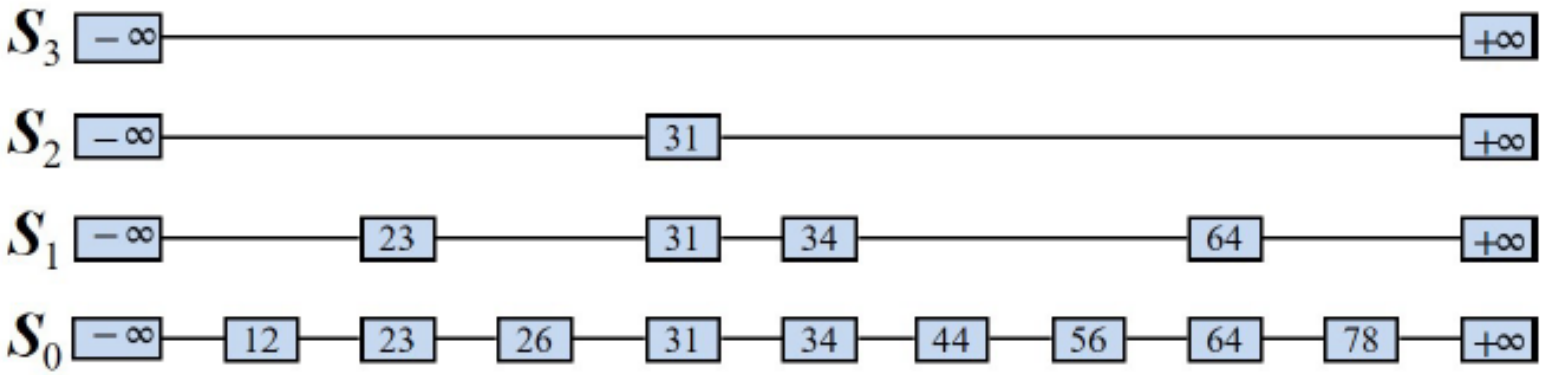
Computer Science 3A

Practical 9

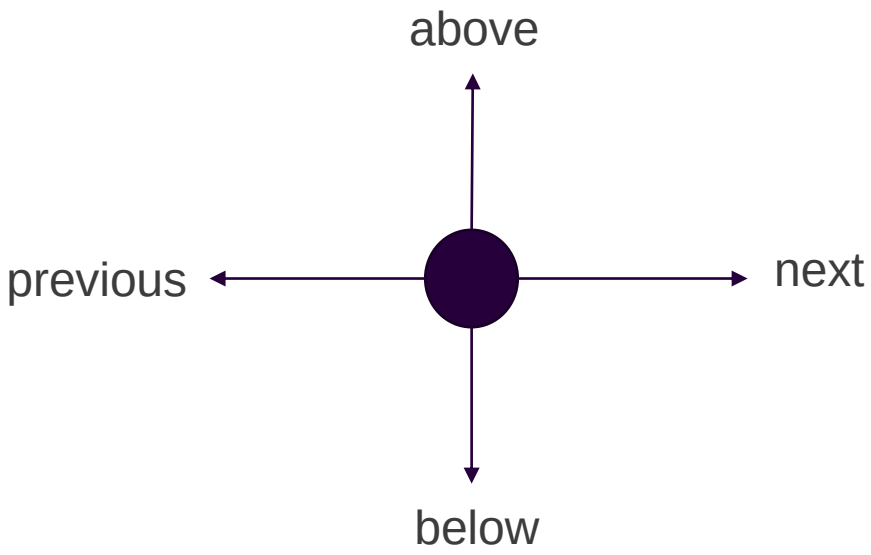
25 April 2024

UNIVERSITY OF JOHANNESBURG

Skip List ADT



Skip List Nodes



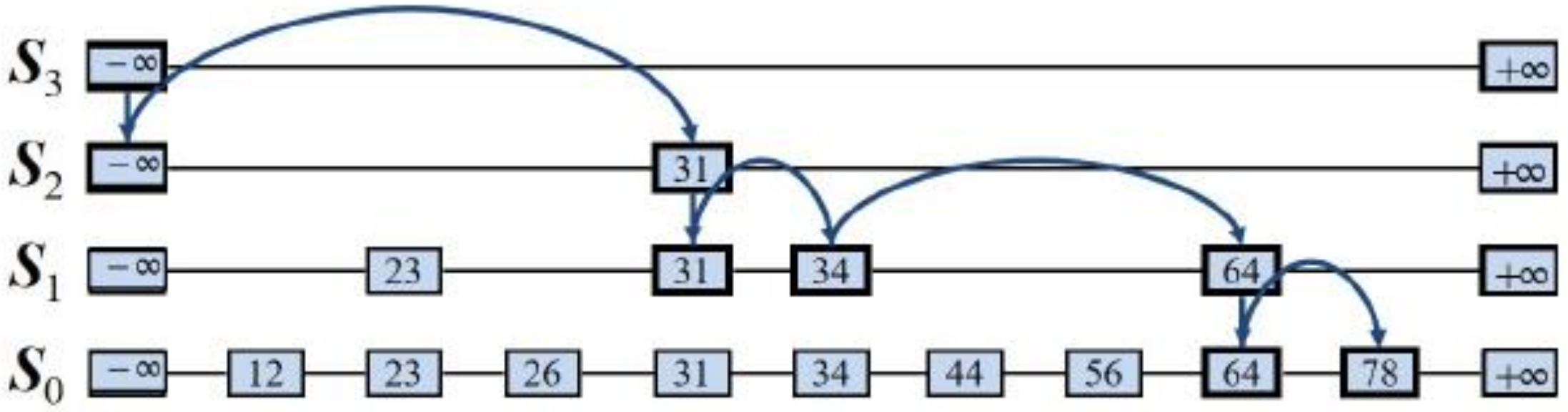
Skip List Nodes

SLNode class to complete:

- References to nodes: next, prev, above, below
- Element
- Constructor
- Accessors
- Mutators

searchSkip (K key)

Example: search for 78



Algorithm: searchSkip (K key)

(In Lecture Slides)

SkipSearch(key*) – return position p in the bottom list such that p has the largest key less than or equal to k*

```
p = startNode
While (p.getBelow() != null) do
    p = p.getBelow()  //drop down
    While (key >= key of p.getNext())
            p = p.getNext() //scan forward
Return p
```


insertAfterAbove(a, b, element)

- Insert after “a” and above “b”
- To be used for insertion in the SkipList

insertAfterAbove(a, b, element)

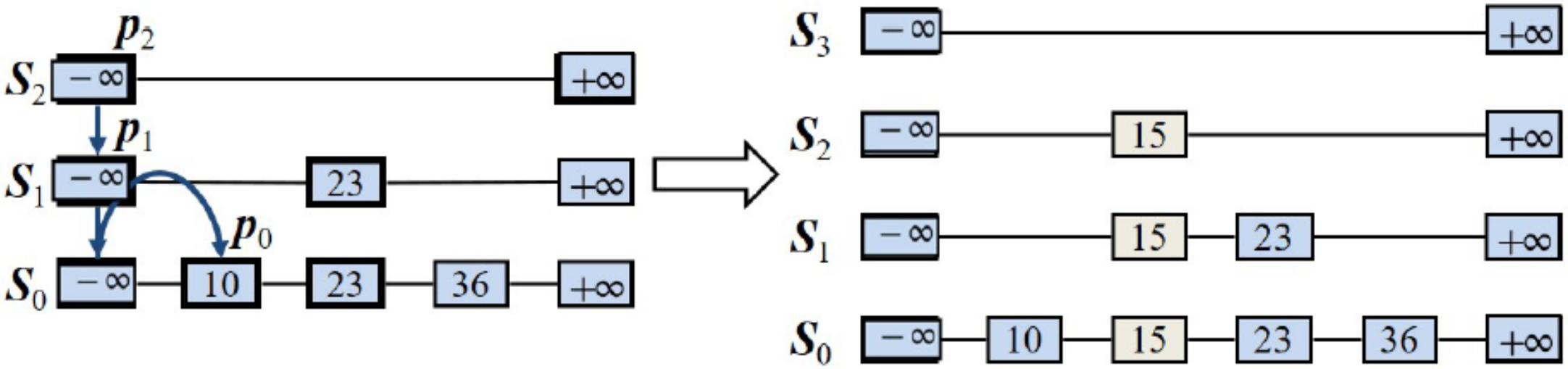
insertAfterAbove(a, b, elem)

get reference to the element after a – “after”
get reference to the element above b – “above”
create new SLNode with parameters:
next, prev, above, below, elem
(after, a, above, b, elem)
update “after’s” previous
update “a’s” next
update “above’s” below
update “b’s” above

*remember to check that the parameters are not null before setting/getting values

Insert(K key, V value) (In Lecture Slides)

Example: insert key 15, with i = 2



Insert(K key, V value) (In Lecture Slides)

```
Insert(k,v)
        p = searchSkip(k)
        newElem = new SkipEntry (k,v)
        SLNode q = insertAfterAbove(p, null, newElem) //at the bottom level
        SLNode t
        towerH = 0
        while (coinflip == heads) //* hint: coinflip with rand.nextInt
                increment towerH
                if (towerH >= height)
                        increment height
                        t = start.getNext
                        start = insertAfterAbove(null, start, SkipEntry(null,null, MIN))
                        insertAfterAbove (start, t, SkipEntry(null, null, MAX))
                while (above(p) == null)
                        p=p.getPrev //scan back
                p = above(p) //jump to upper level
                q = insertAfterAbove(p,q, newElem) // add to the tower of new entry
        increment size and return newElem
```


Iterator<IEntry<K,V>> Entries()

```
SLNode current = start
While (below(current) != null) //go to the lowest level
        current = below(current)
current = next(current)

//add all elements to the positionList until we hit the max
positionList = new PositionList
While (current.element.getSentinel != MAX)
        add to the positionlist
        progress to the next node

Return the iterator
```