

Ordered Collection ADT + Search Trees

Note Title

10/12/2007

Conceptual view $\langle a, a, f, g, r, t, v, w, y \rangle$

methods

Iteration is sorted

most
data
structures
for
ordered
collection
ADT
support

add, locate/search, remove

min in ex a

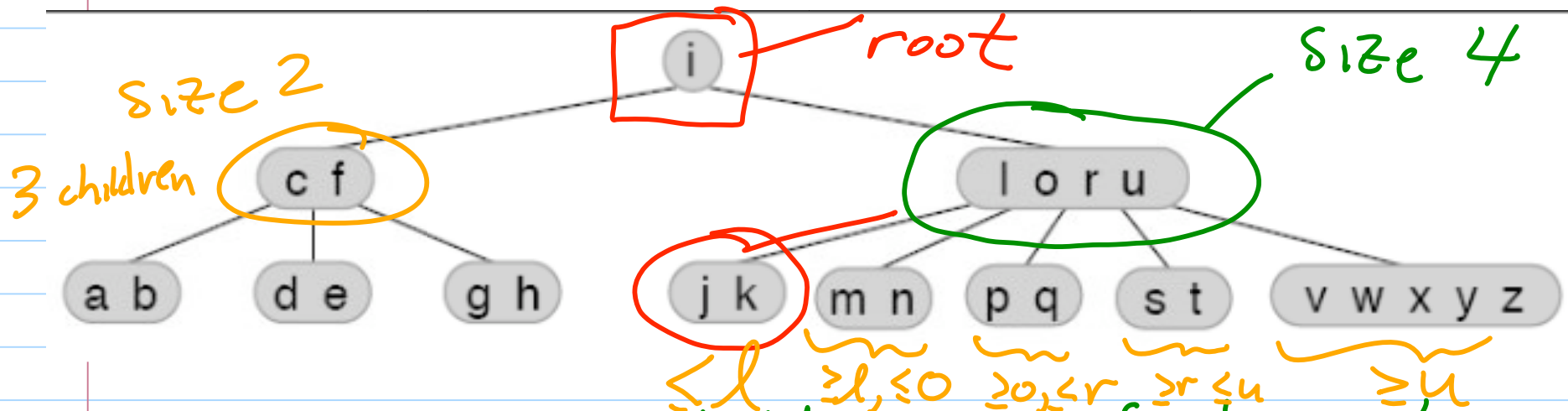
max in ex y

Successor in ex successor(k) is v

predecessor in ex predecessor(k) is g

support these in logarithmic time

Abstract Search Tree

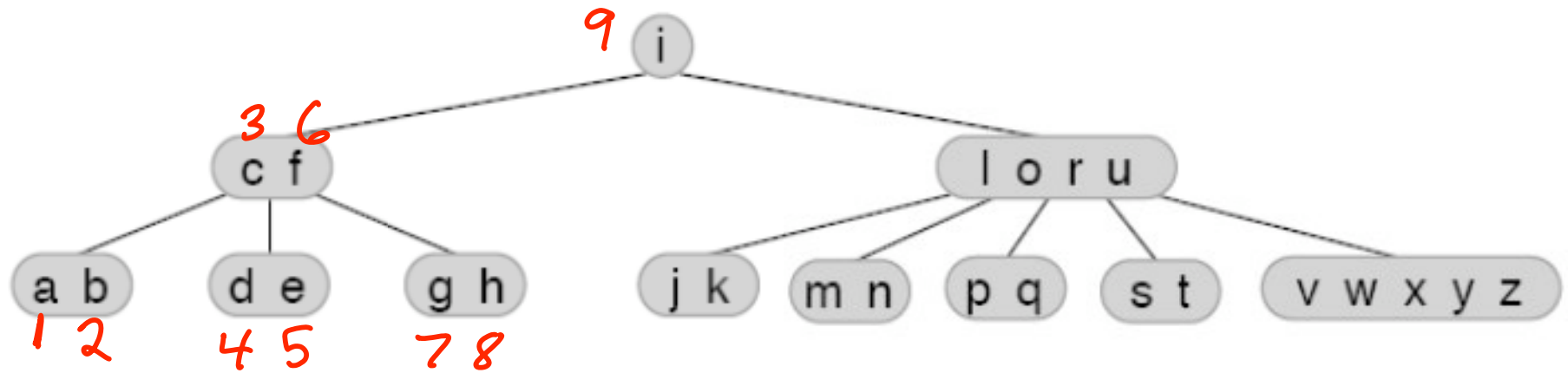


each node can hold any # of elements

Let s be the # elements (size) of a node

Have $s+1$ children (possibly empty)

In order-traversal - Sorted order
linear time



visit(x)

visit(x, child(0))

output element 0

visit(x, child(1))

output element 1

visit(x, child(s))

} loop

Representation Property

INORDER

binary search
tree

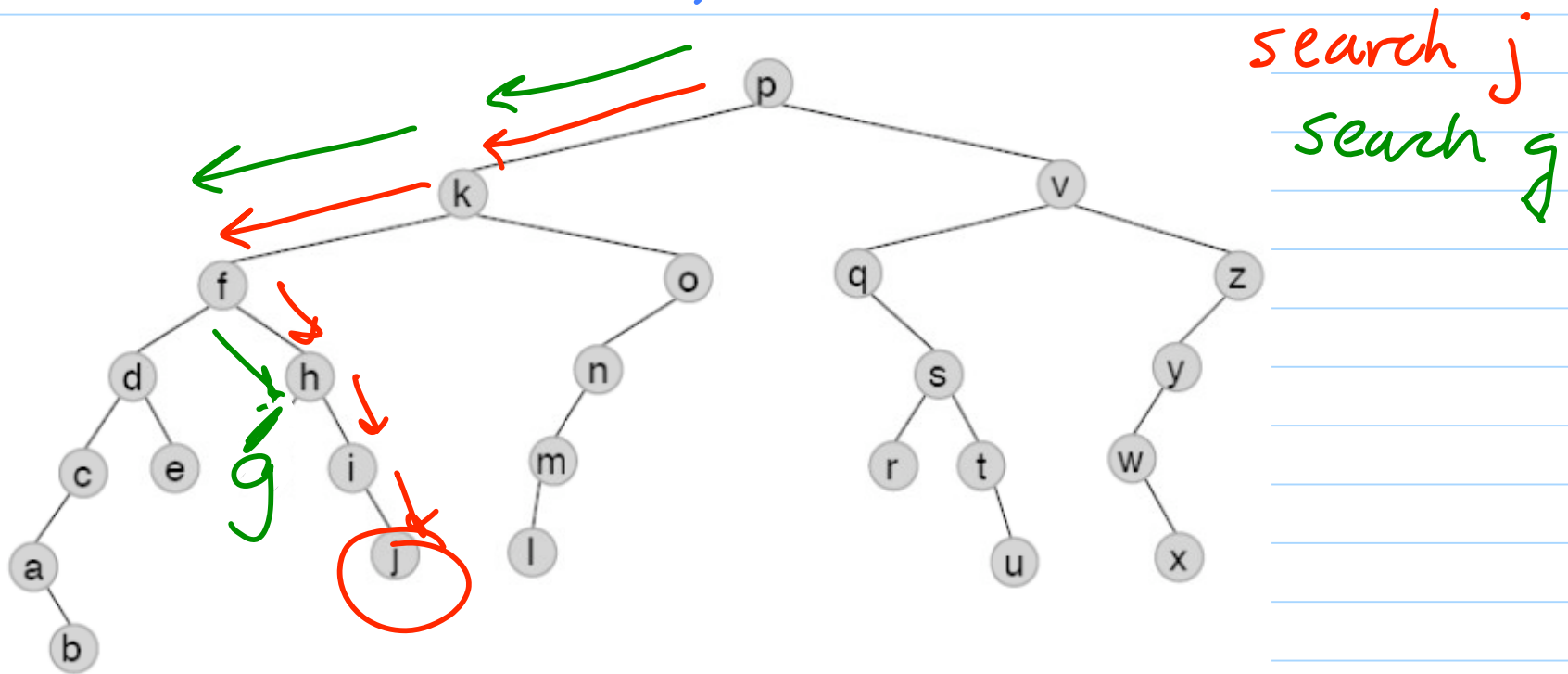
$$T(x.\text{Left}) \leq x.\text{data} \leq T(x.\text{right})$$

node

all items in left subtree

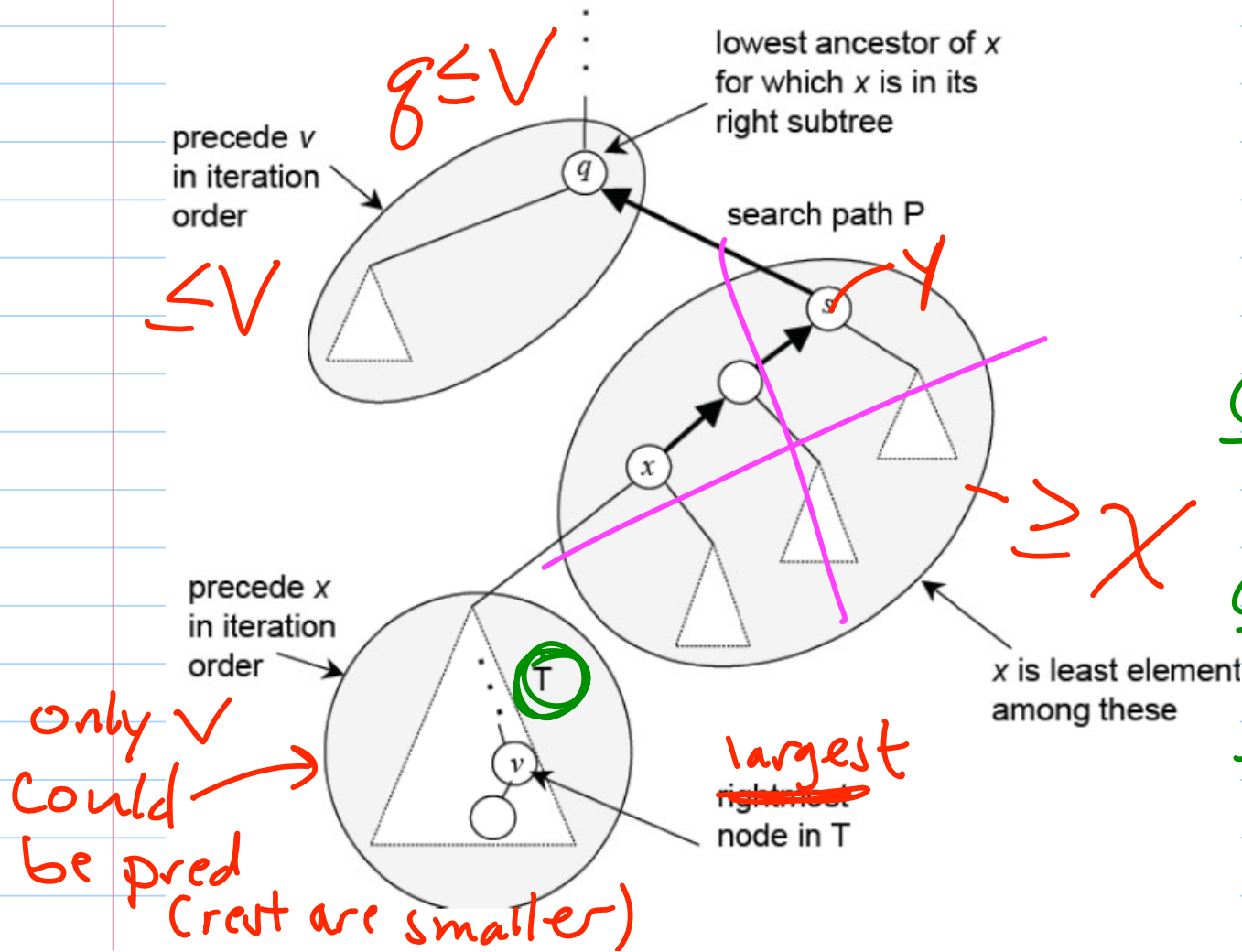
$$T(x.\text{child}(0)) \leq x.\text{data}(0) \leq T(x.\text{child}(1)) \leq x.\text{data}(1) \\ \leq \dots \leq x.\text{data}(s-1) \leq T(x.\text{child}(s))$$

Review binary search trees



insertion is a search that ends at "null"
+ place it.

Finding predecessor



Note

$$g \leq v \leq x \leq s$$

Find pred. of x

Case 1 T exists
 v is pred

Case 2 T empty
 $\& g$ exists
then g is pred.

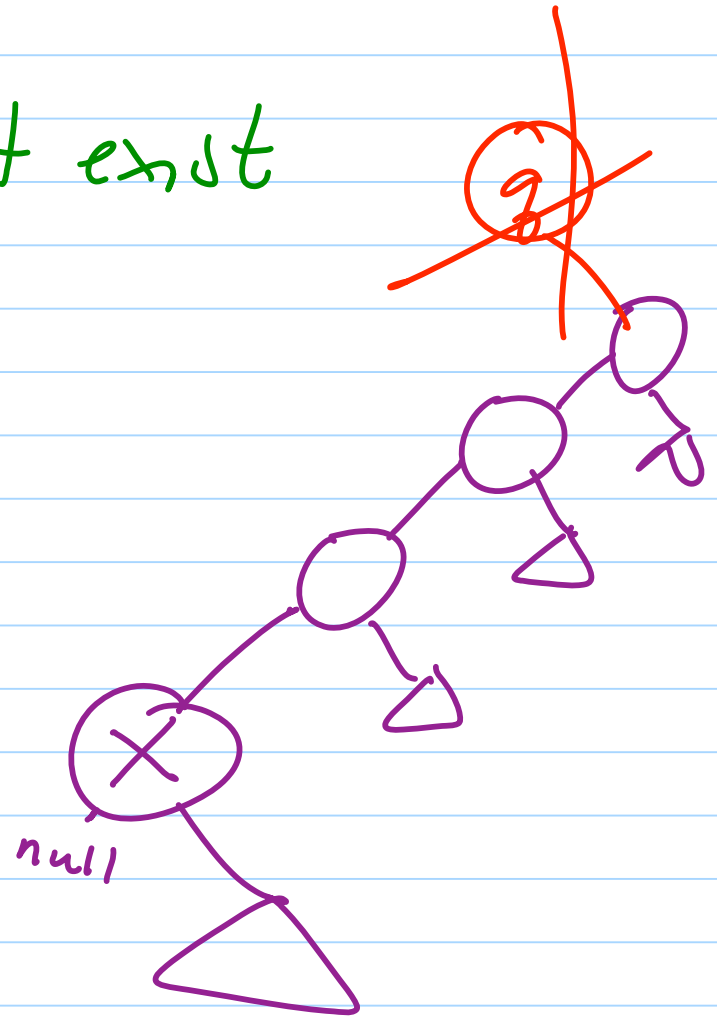
Case 3.

T empty + q doesn't exist

No predecessor

x is first
in an in order
traversal

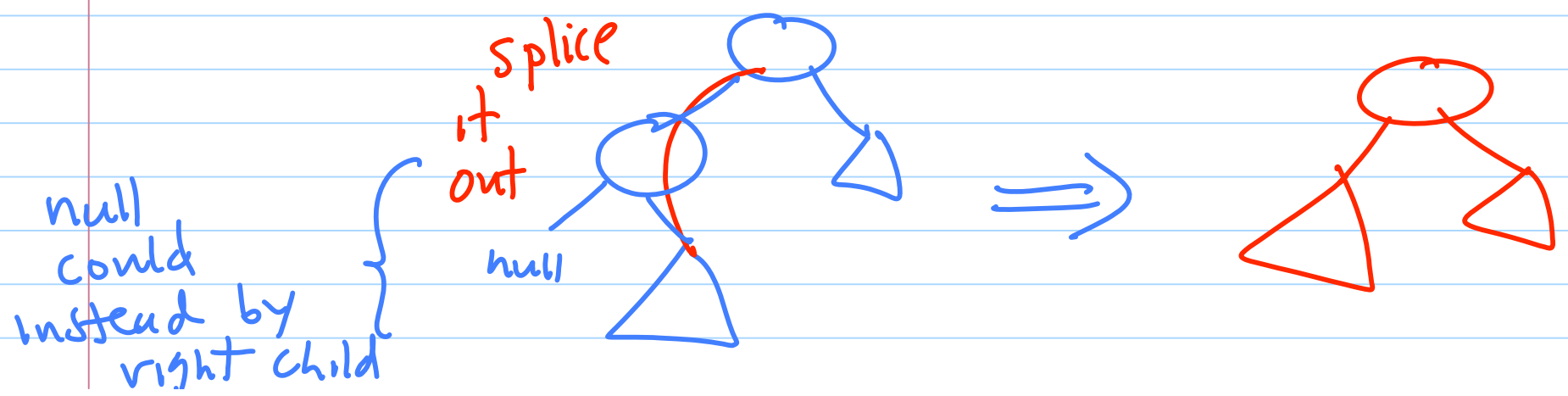
Successor is symmetric



Delete

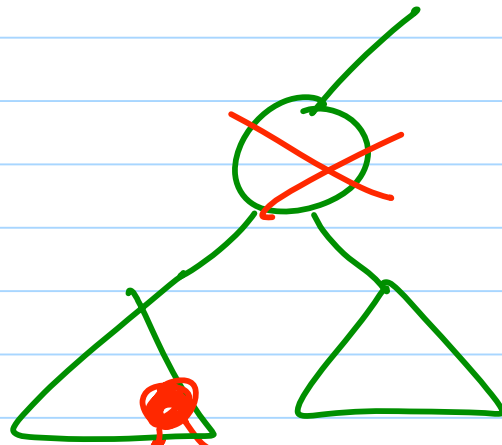
easy case - delete a leaf
remove it

medium case delete a node with
one child

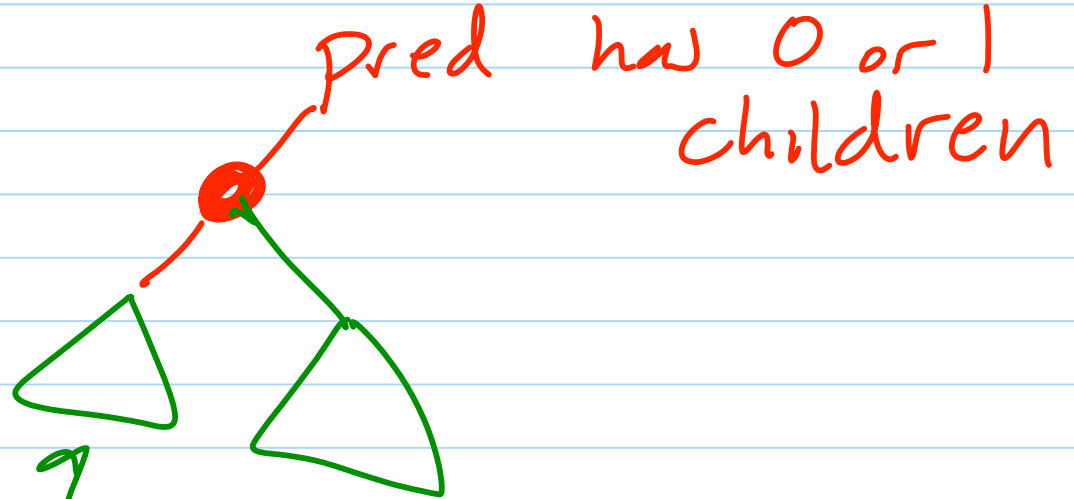


Hardest case delete a node
with 2 children

could use succ



pred
is max
in subtree



remove pred from left subtree