

Zip Trees

Robert Tarjan and **Caleb Levy**

Princeton University
& Intertrust Technologies

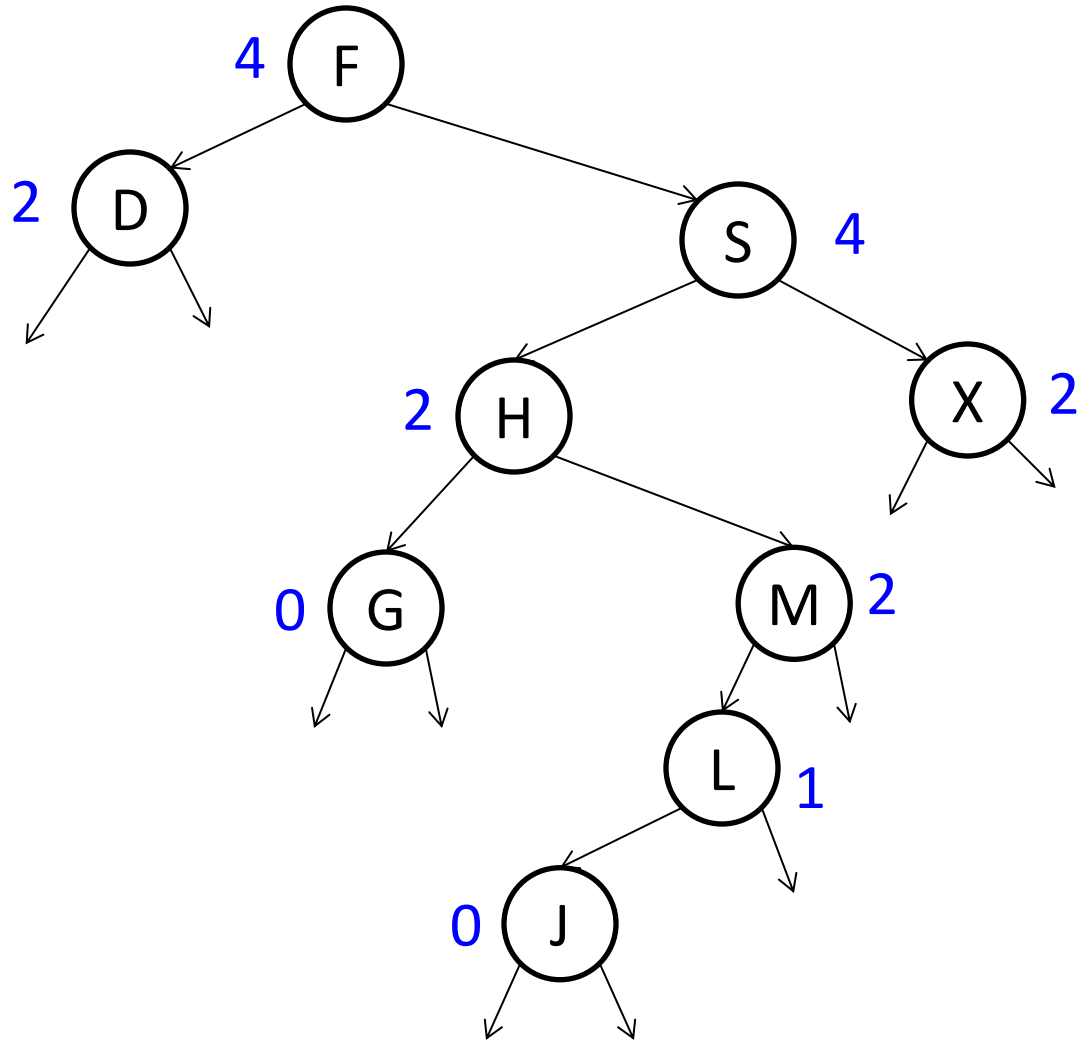
HALG 2018

Zip Trees

A Binary Search Tree where:

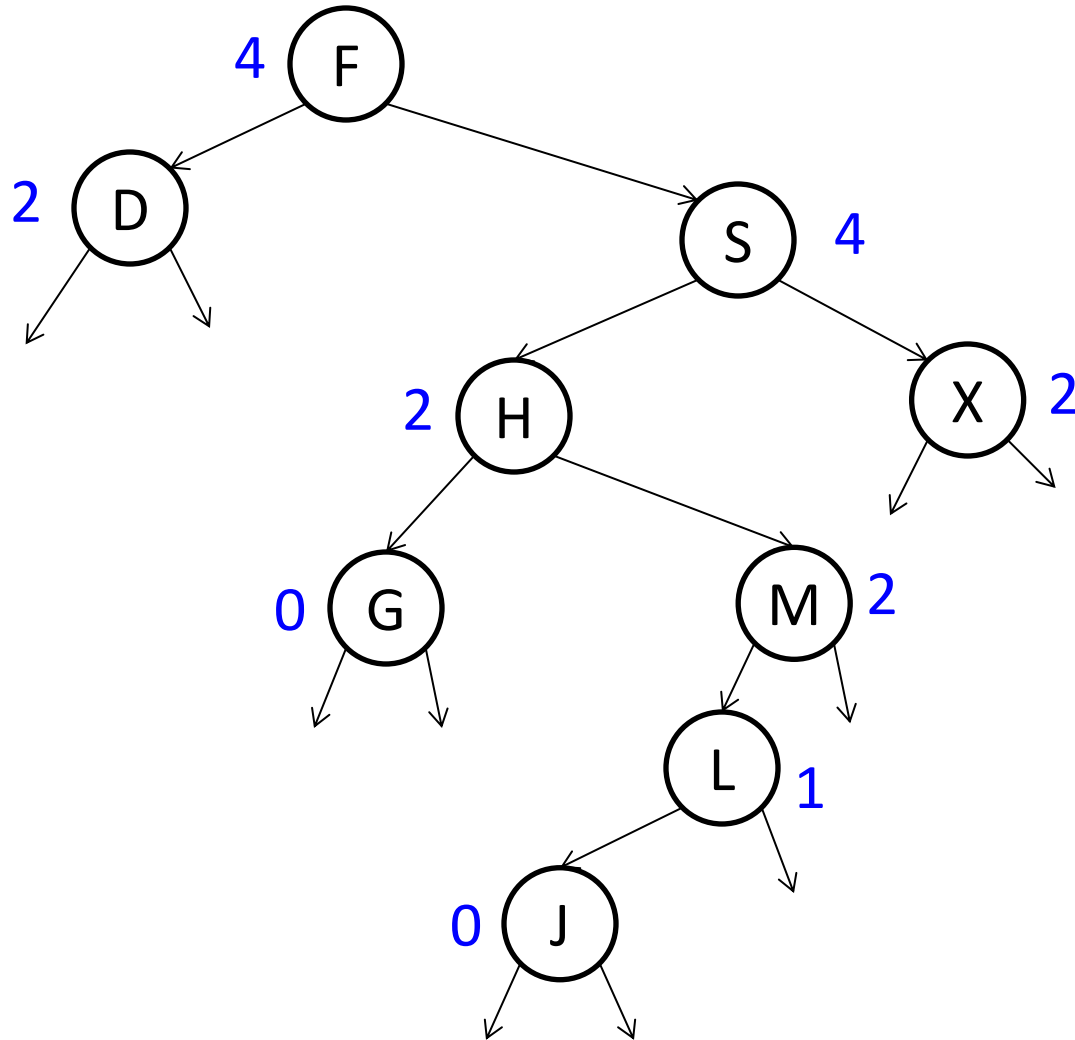
- Keys have random **geometrically** distributed ranks
 $Pr[\text{rank}(x) = k] = 1/2^{k+1}$
- Constructed recursively:
root(T) is the **smallest node of maximum rank**
- Variant of **Treaps**, equivalent to **Skip Lists**

A Zip Tree



INSERTION

Insertion by Unzipping



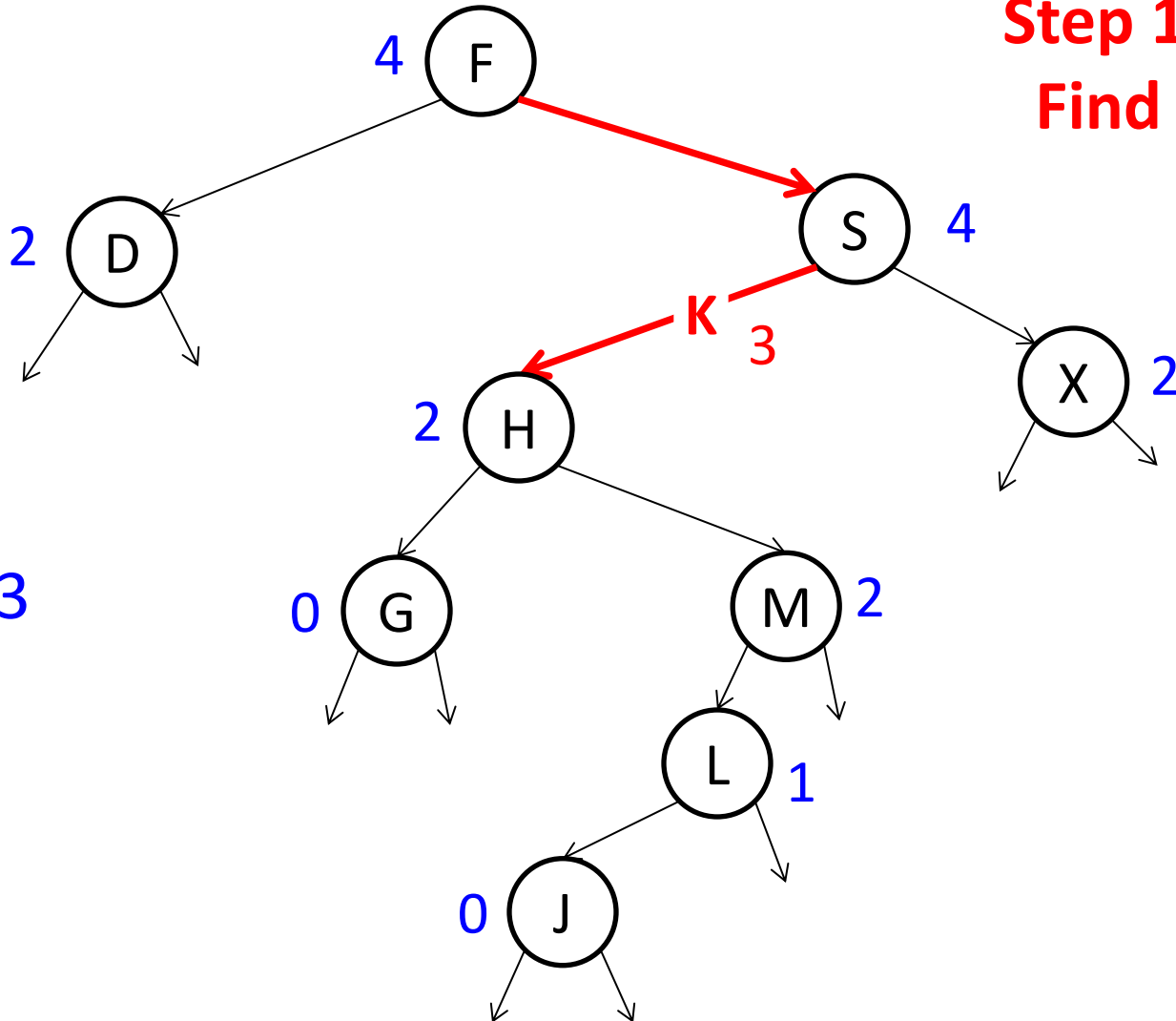
Insert:

Key = K

Rank = 3

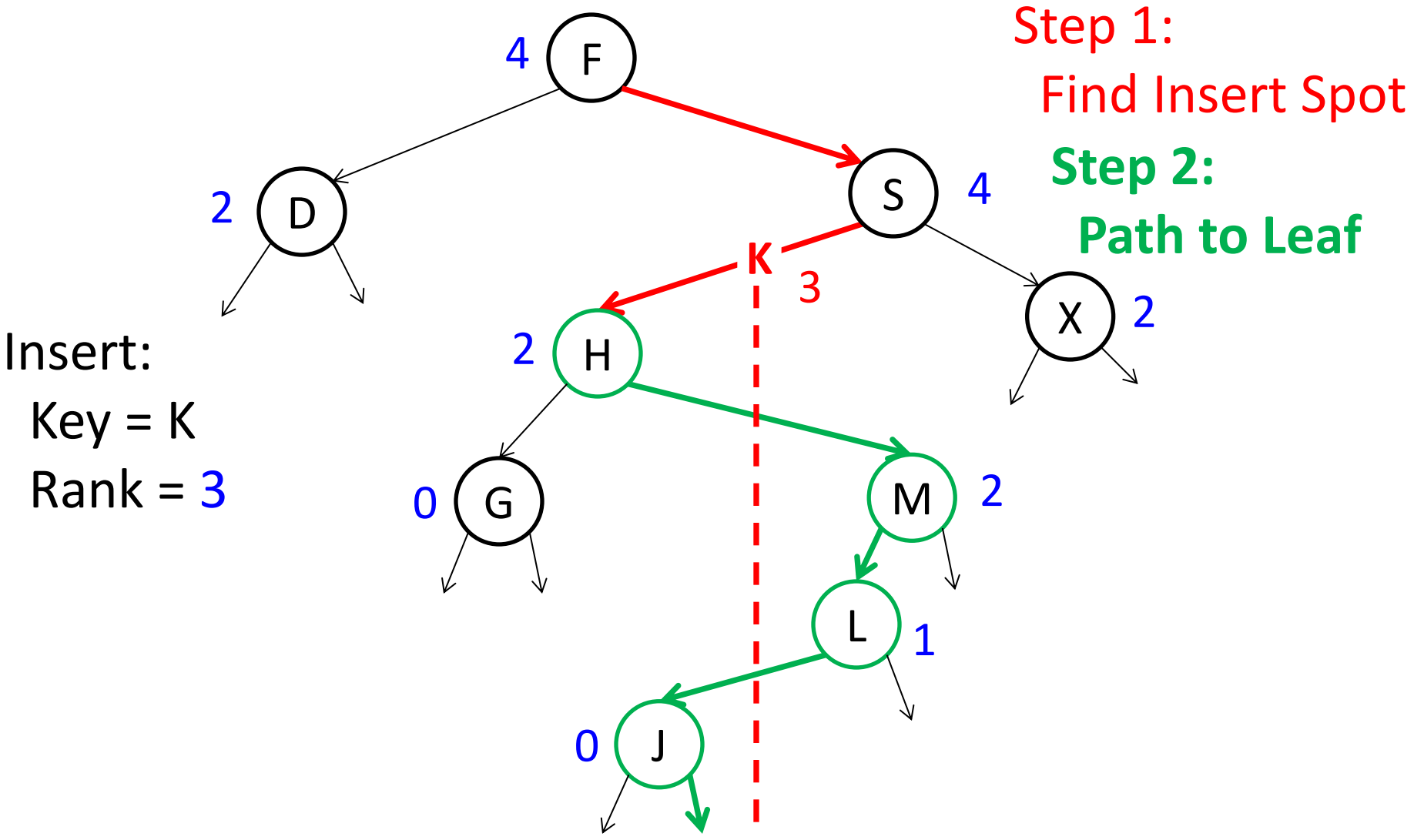
Insertion by Unzipping

Step 1:
Find Insert Spot

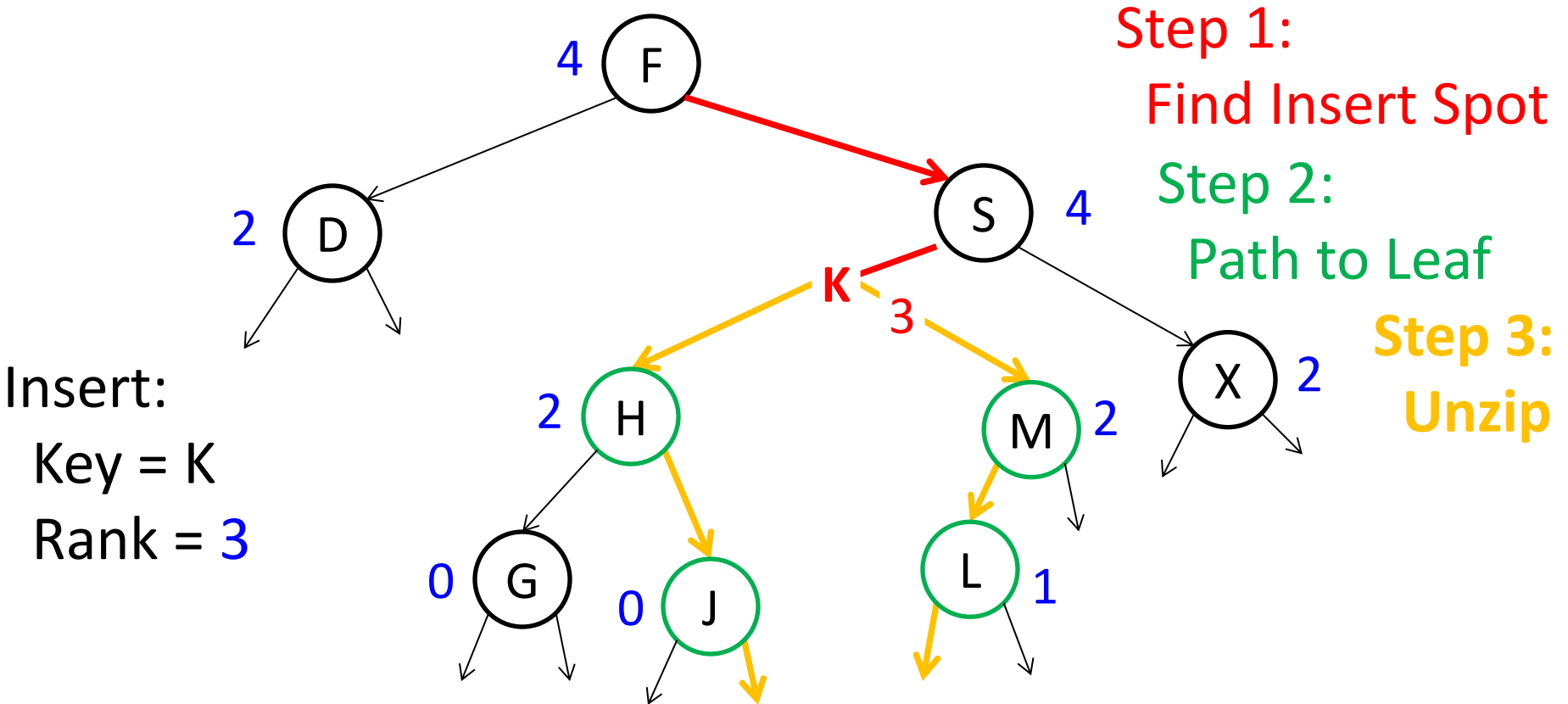


Insert:
Key = K
Rank = 3

Insertion by Unzipping



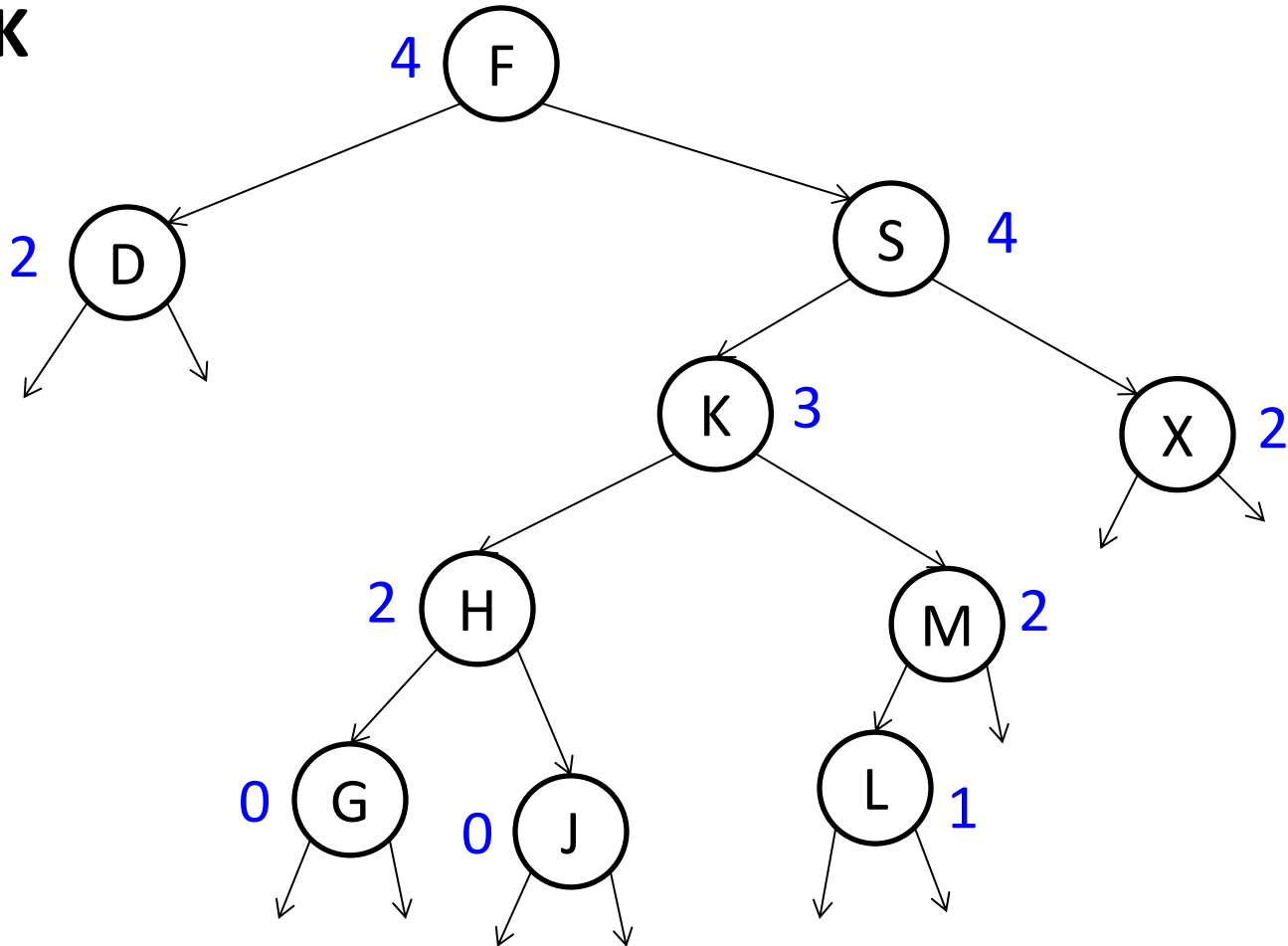
Insertion by Unzipping



DELETION

Deletion by Zipping

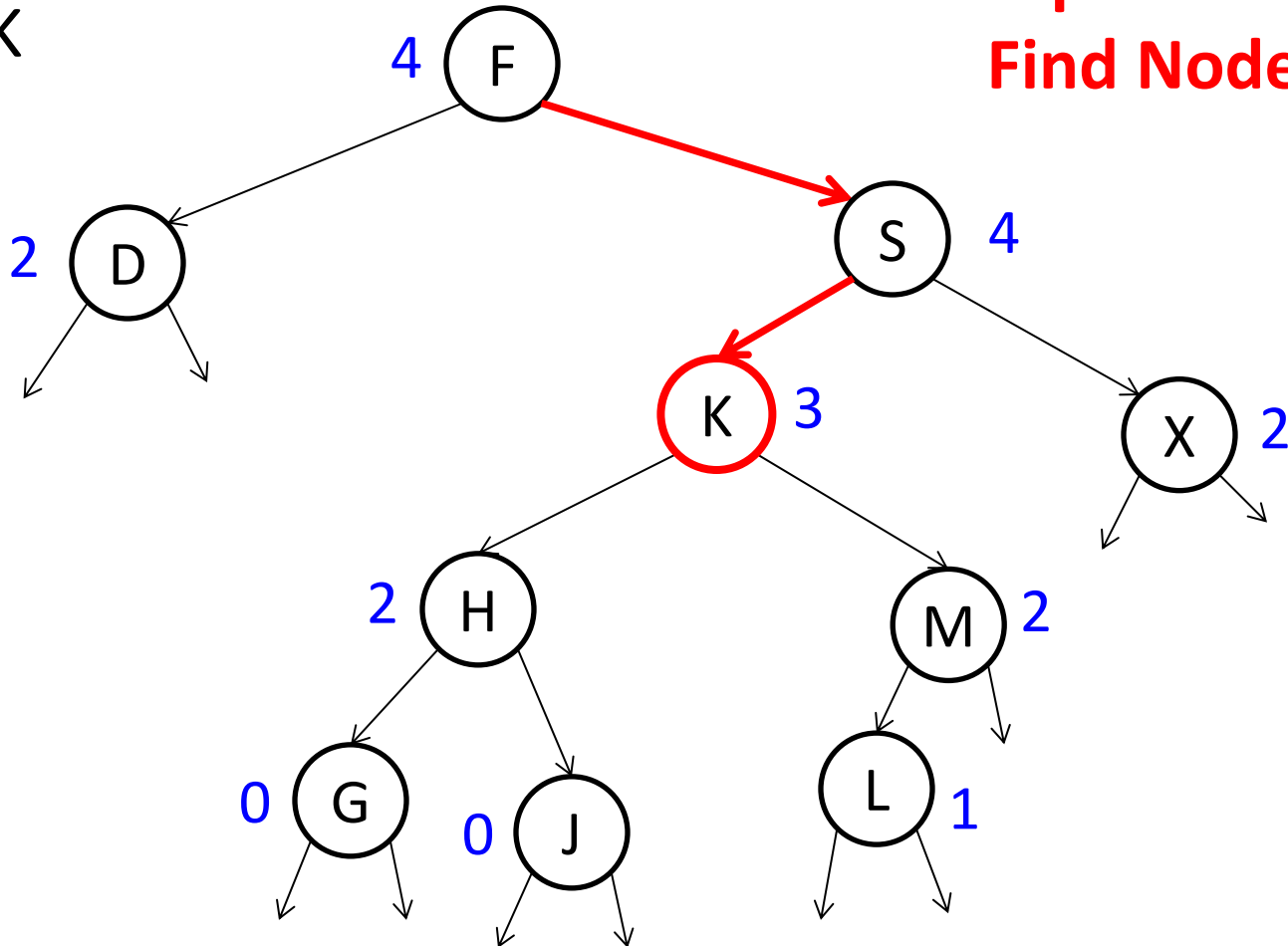
Delete:
Key = K



Deletion by Zipping

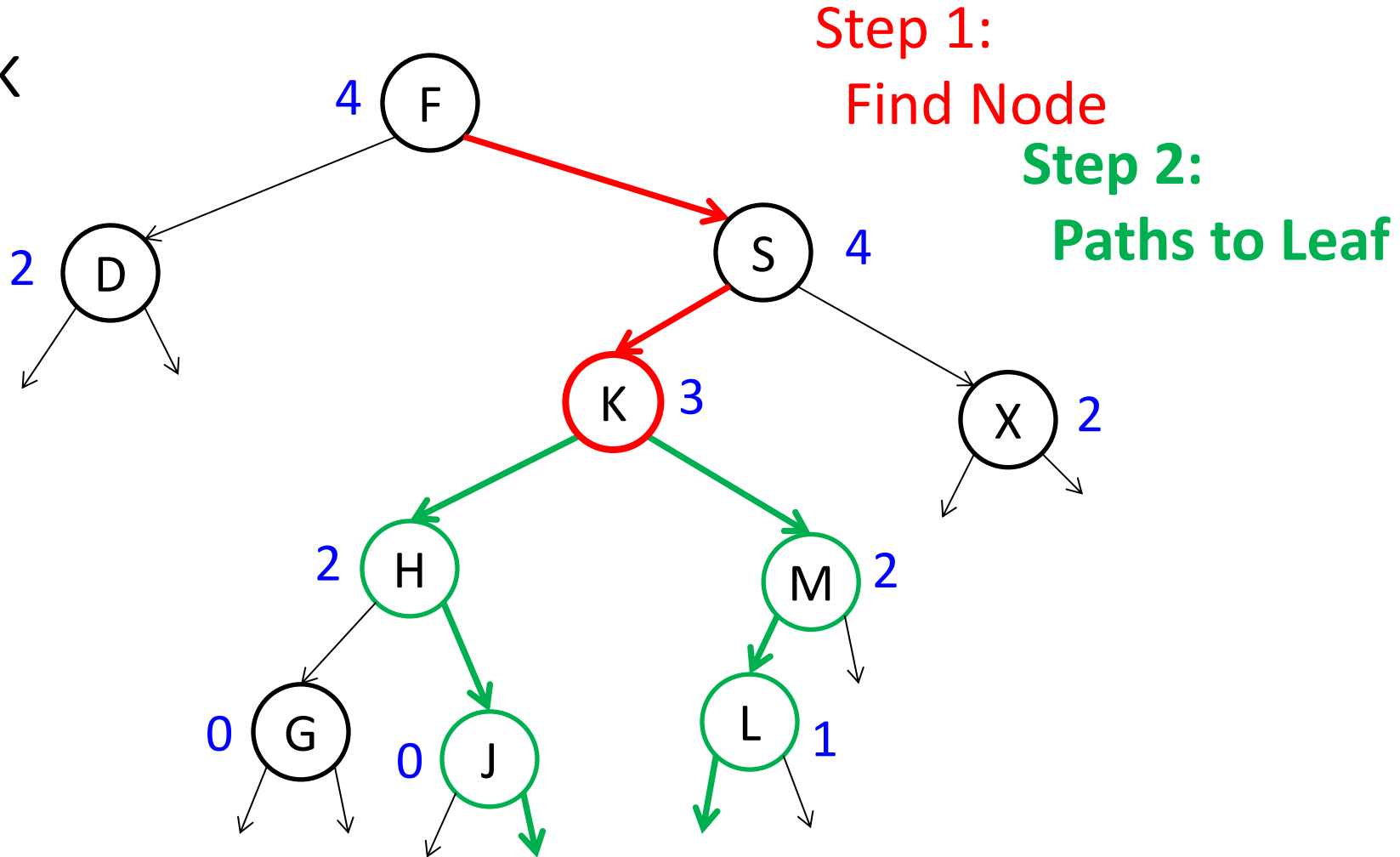
Delete:
Key = K

Step 1:
Find Node



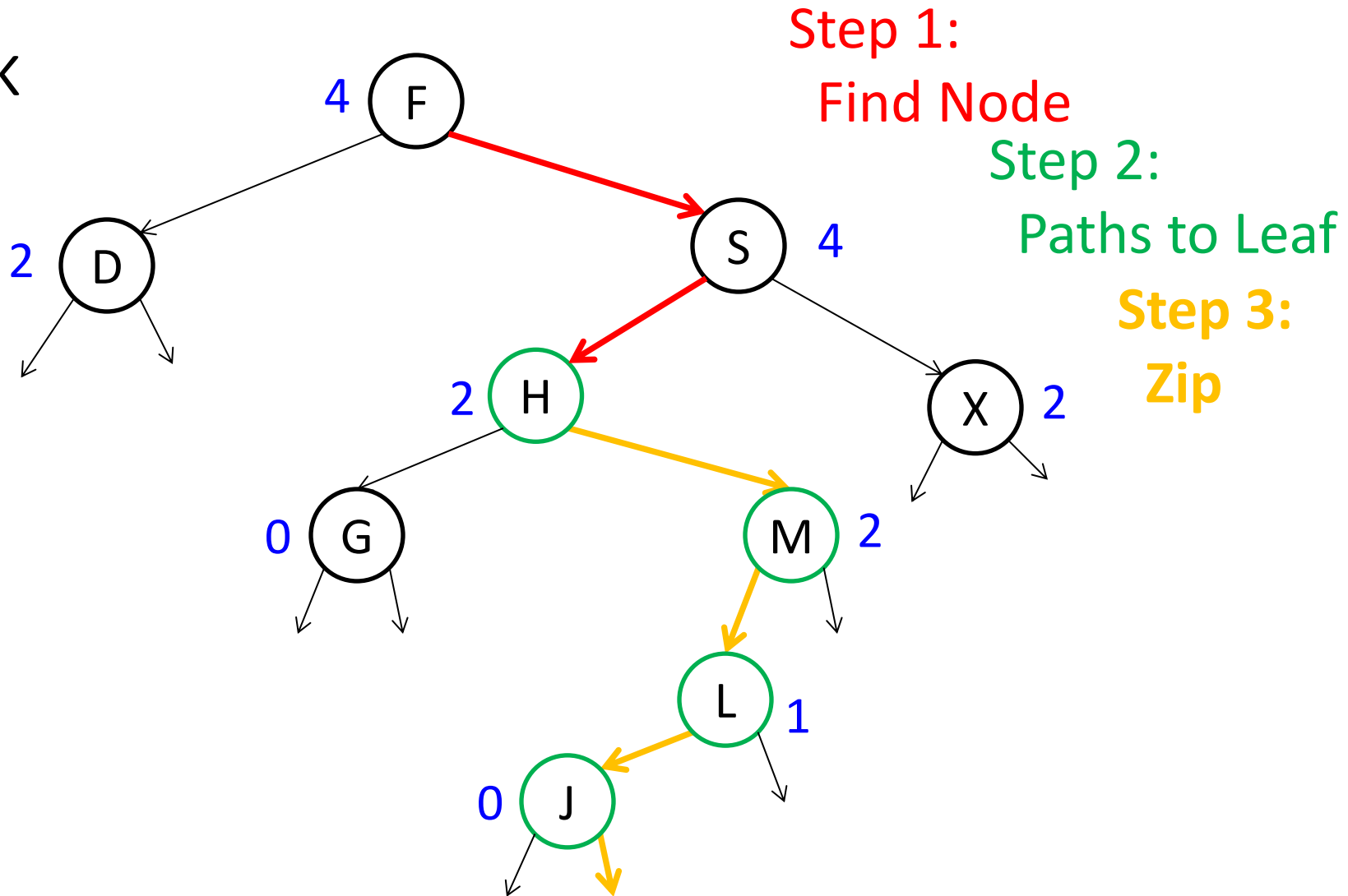
Deletion by Zipping

Delete:
Key = K



Deletion by Zipping

Delete:
Key = K



Properties of Zip Trees

- Tree depth is $O(\log n)$ W.H.P.
- A **unique** tree for each set of *keys* and *ranks*
- $O(1)$ pointer changes per Insert/Delete
- Insert & Delete purely **top-down**, no **rotations**
- W.H.P. need only $O(\log \log n)$ bits/rank

Current work

Develop, analyze, and implement efficient non-blocking **concurrent** *zip* trees

Related Work

C. Aragon and R. Seidel, “Randomized search trees,” *Algorithmica*, (1996).

C. Martinez and S. Roura, “Randomized binary search trees,” *ACM*, (1998).

W. Pugh, “Skip lists: a probabilistic alternative to balanced trees,” *Comm. ACM*, (1990).

R. Sprugnoli, “Randomly balanced binary trees,” *CALCOLO*, (1980).

C. Stephenson, “A method for constructing binary search trees by making insertions at the root,” *IJCIS*, (1980).

Thanks!