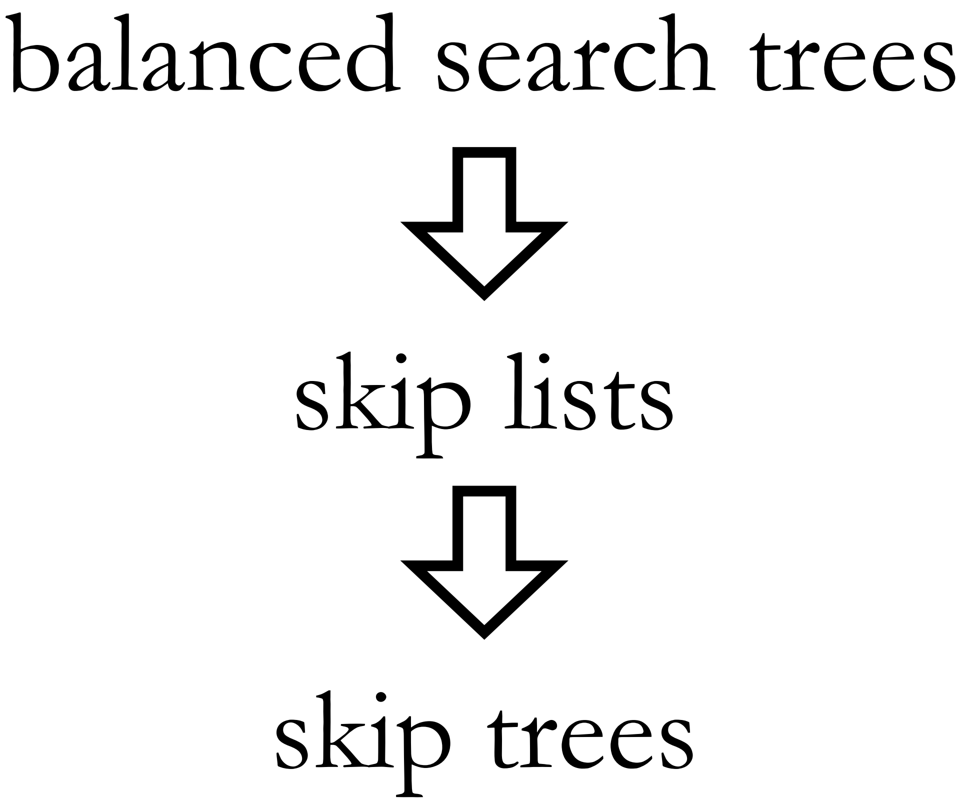
Concurrent Implementation of Skip Trees

Vladimir Magdin
Feb 7 2011

Introduction

topic: alternate balanced tree data structure

outline:



balanced search trees

skip lists

skip trees

Balanced Search Trees

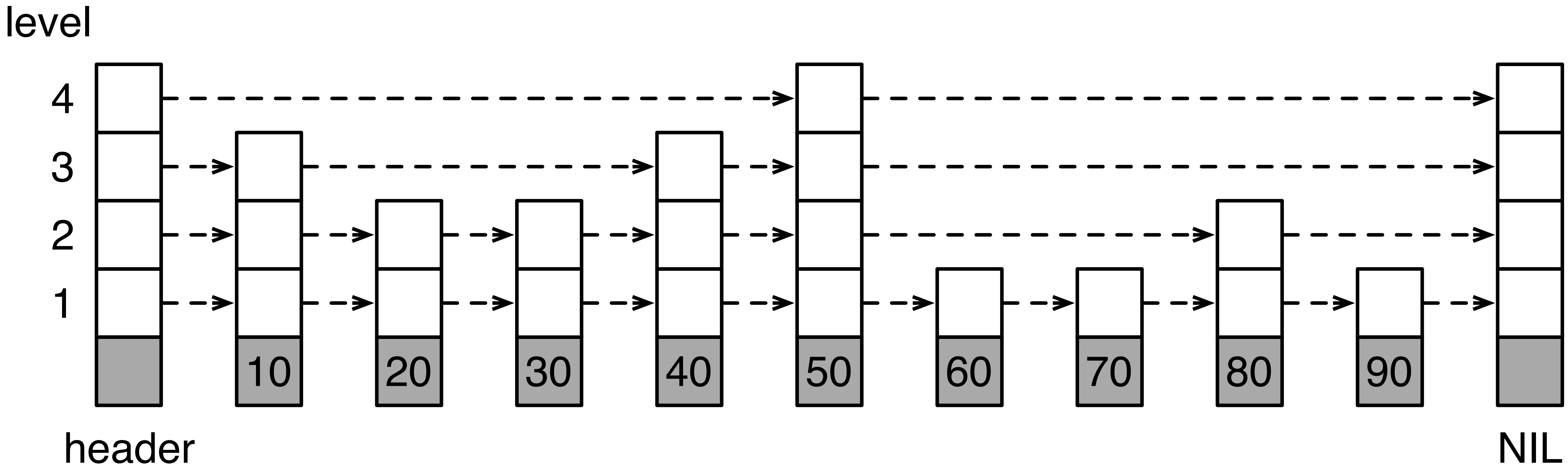
- can represent ordered sets and dictionaries
- O(log(n)) for search, insert, delete operations
- two major types:
    - rotations of nodes (e.g. AVL tree)
    - splitting/merging of nodes (e.g. B-tree)

Balanced Search Trees

disadvantages

- rotations are complicated
    - implementation
    - constant factor: O(c log(n))
- order of insertions might matter

Skip Lists



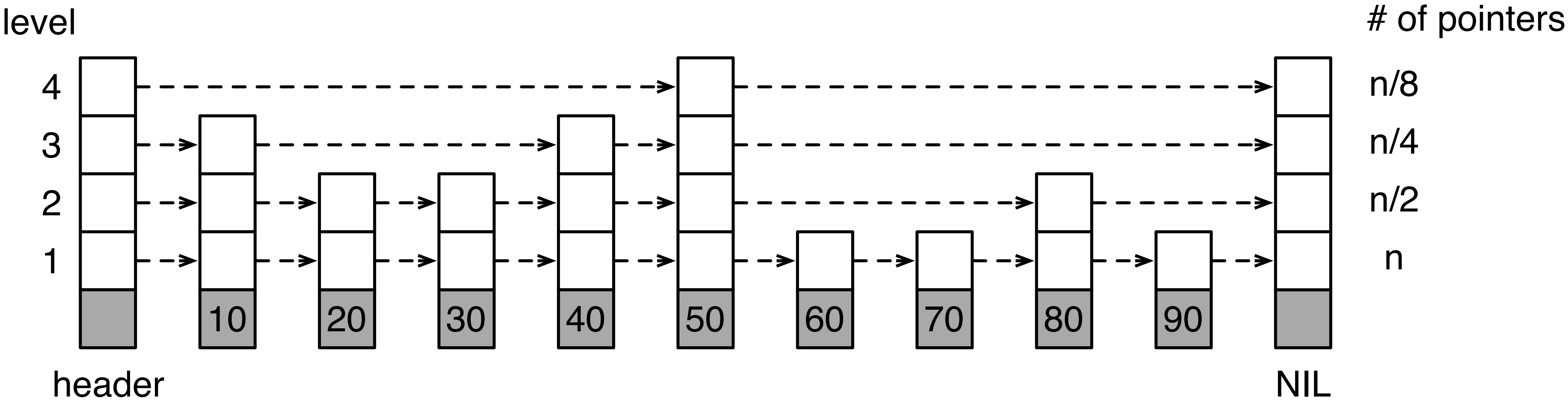
- described by William Pugh in 1990

- hierarchy of lists with different degrees of connectivity

- levels of newly-inserted nodes are chosen randomly

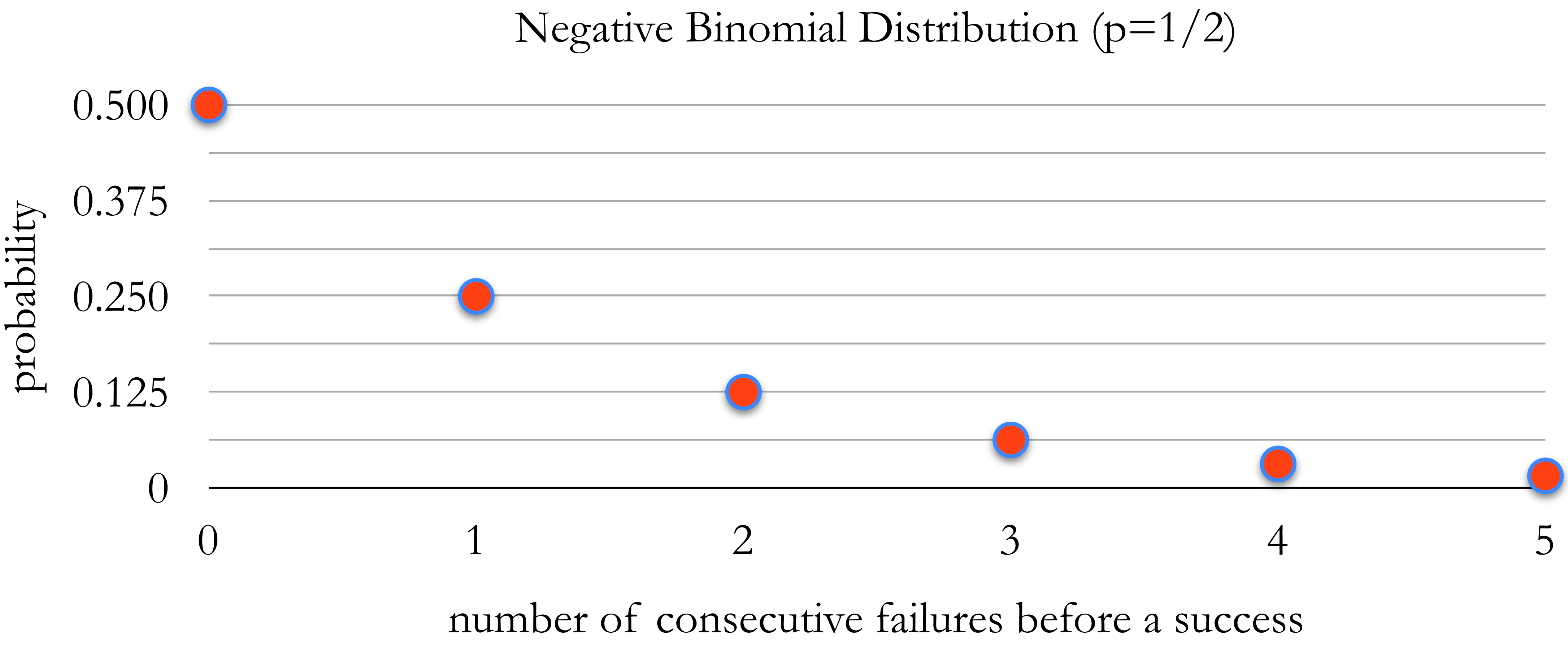
Node Level Selection

- fraction p (e.g. 1/2) of the nodes at level(i) also have pointers at level(i+1)

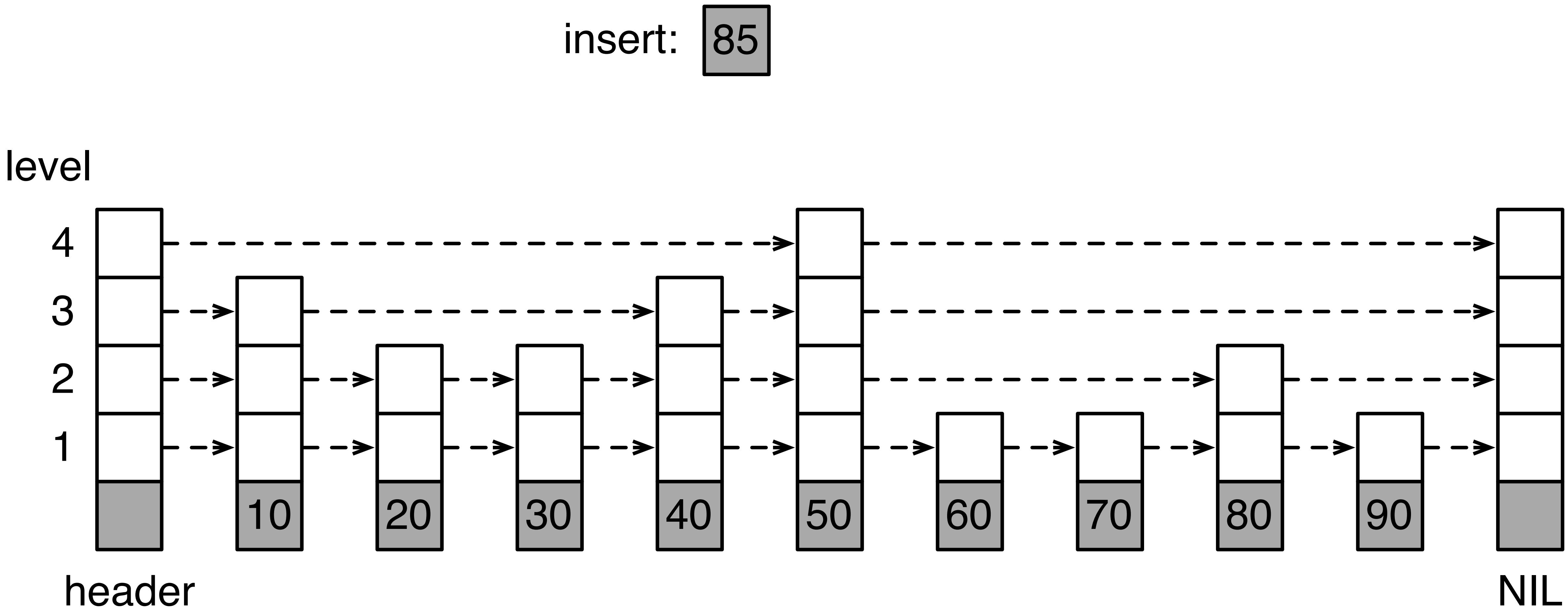


Node Level Selection

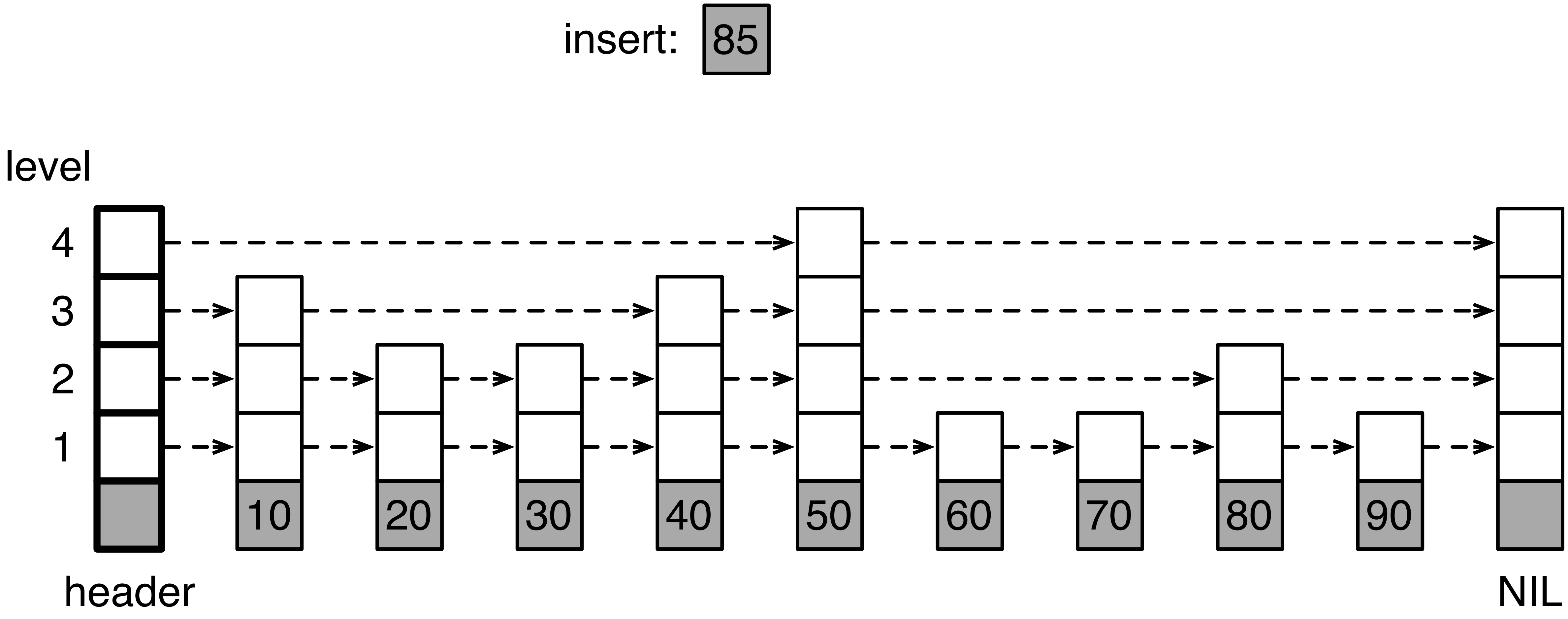
- drawing level from the negative binomial distribution NB(1,p) leads to O($\log_{1/p}$(n)) search



Skip List - Insertion

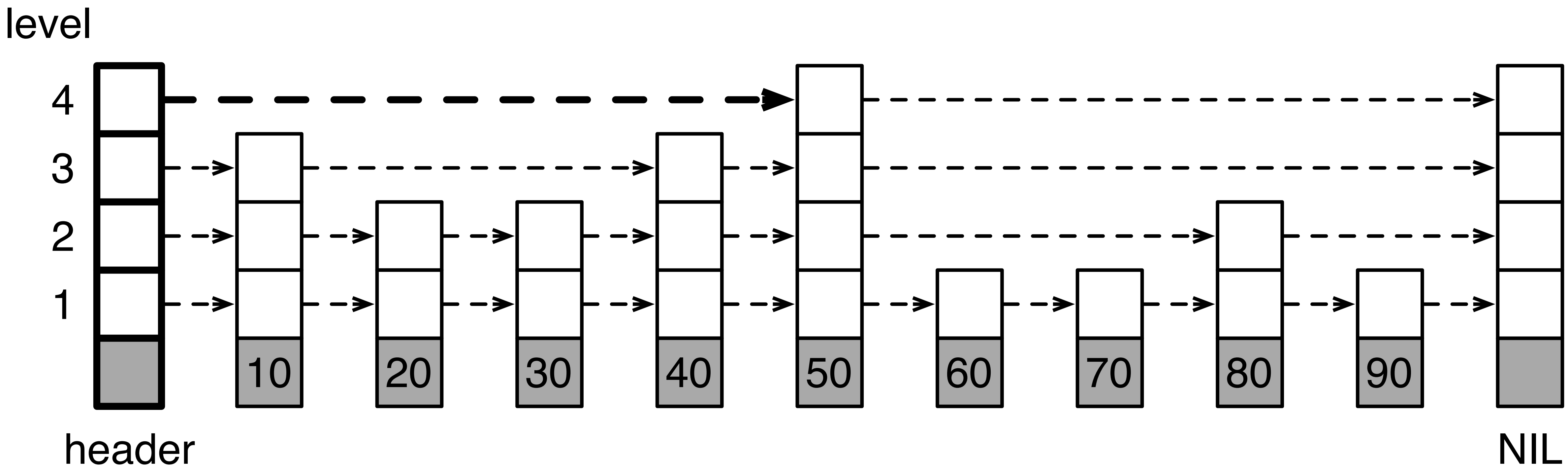


Skip List - Insertion

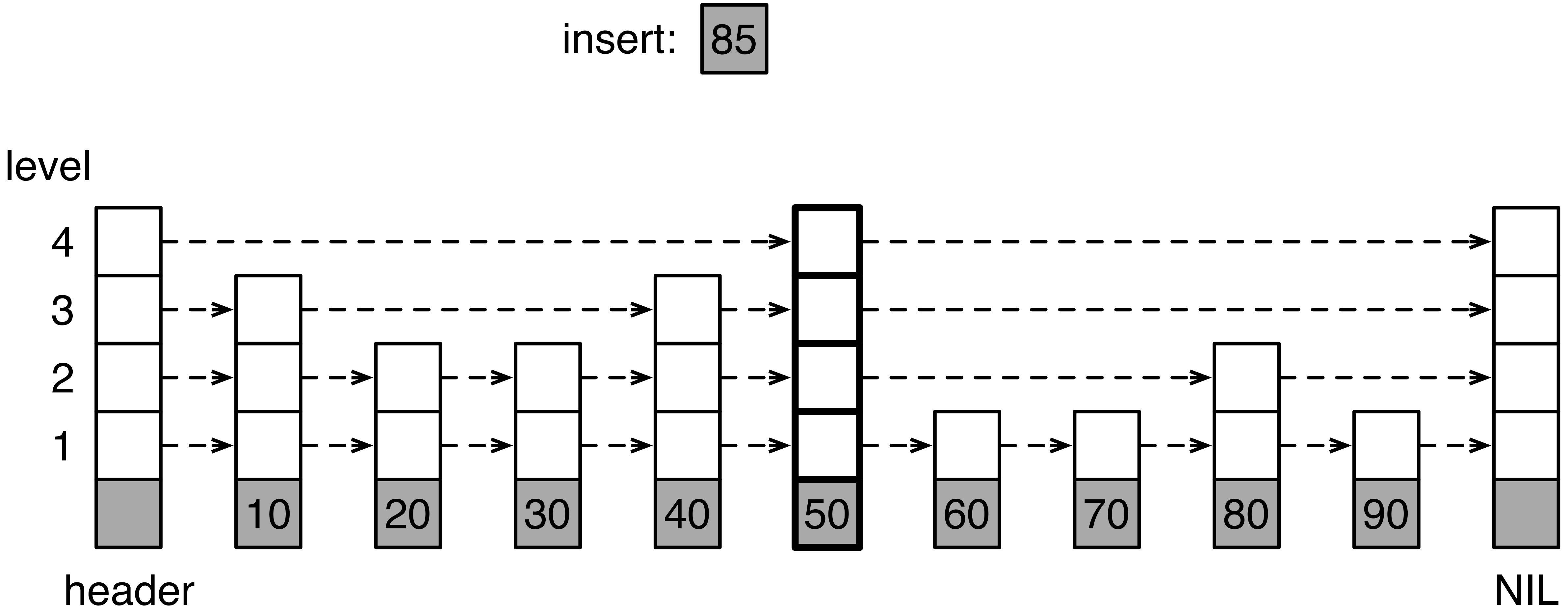


Skip List - Insertion

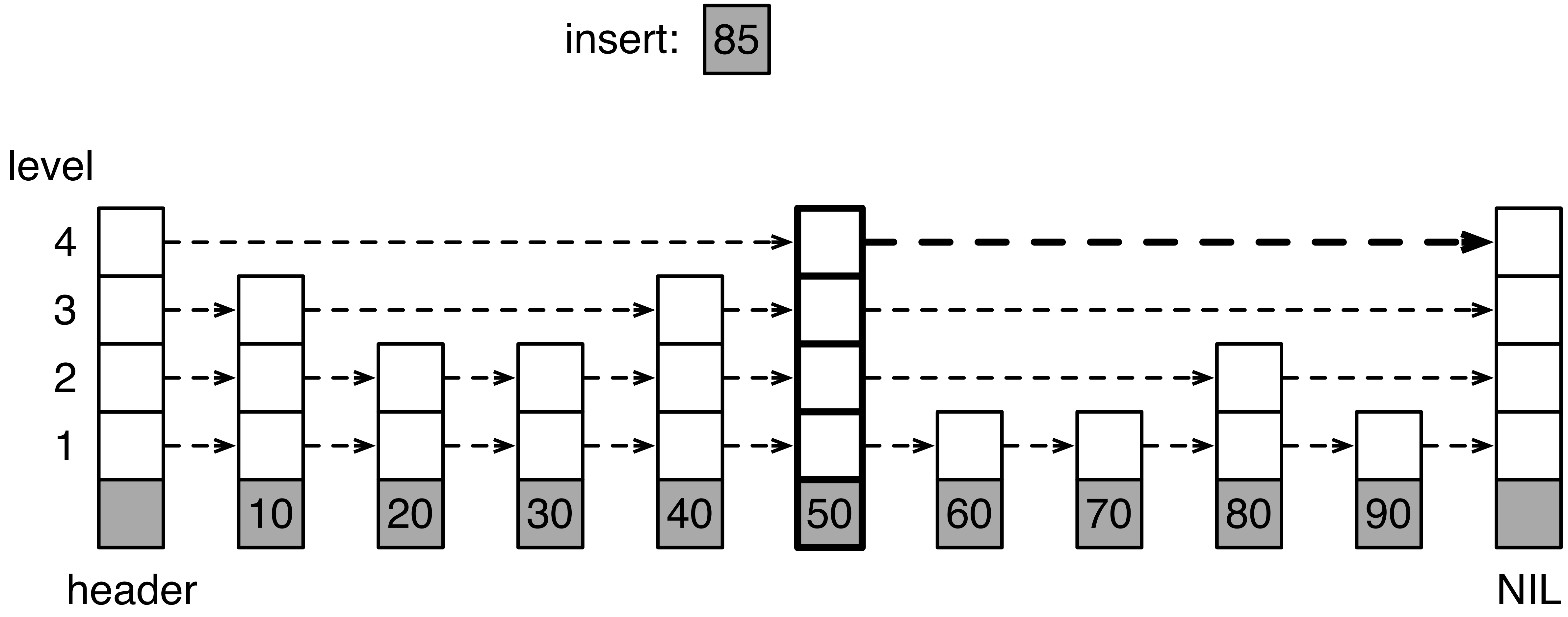
insert: 85



Skip List - Insertion

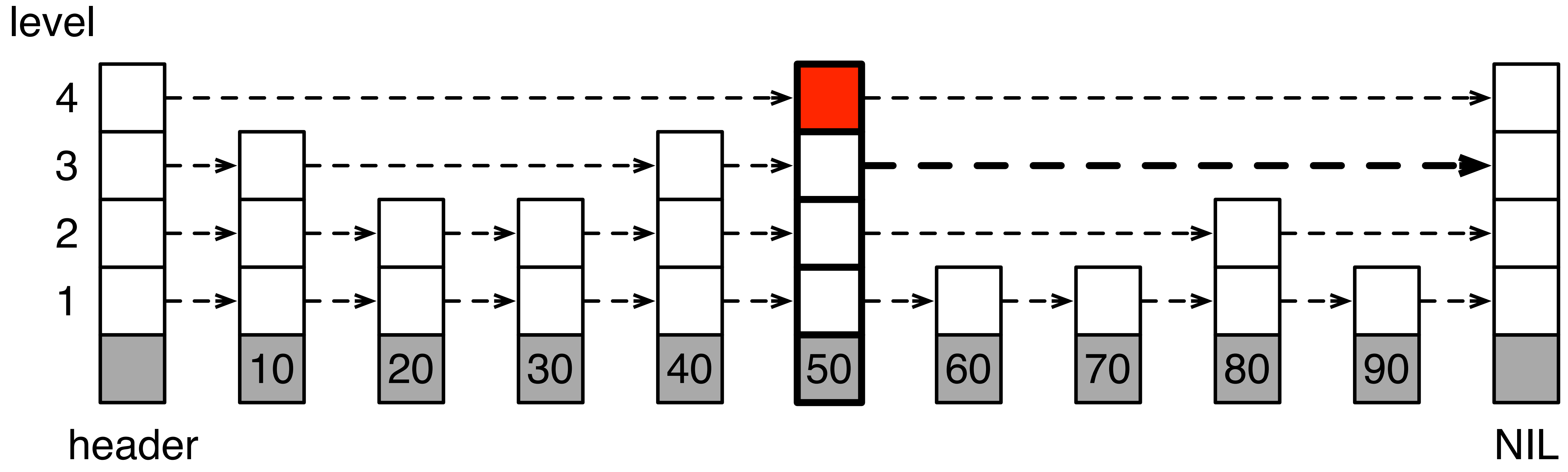


Skip List - Insertion

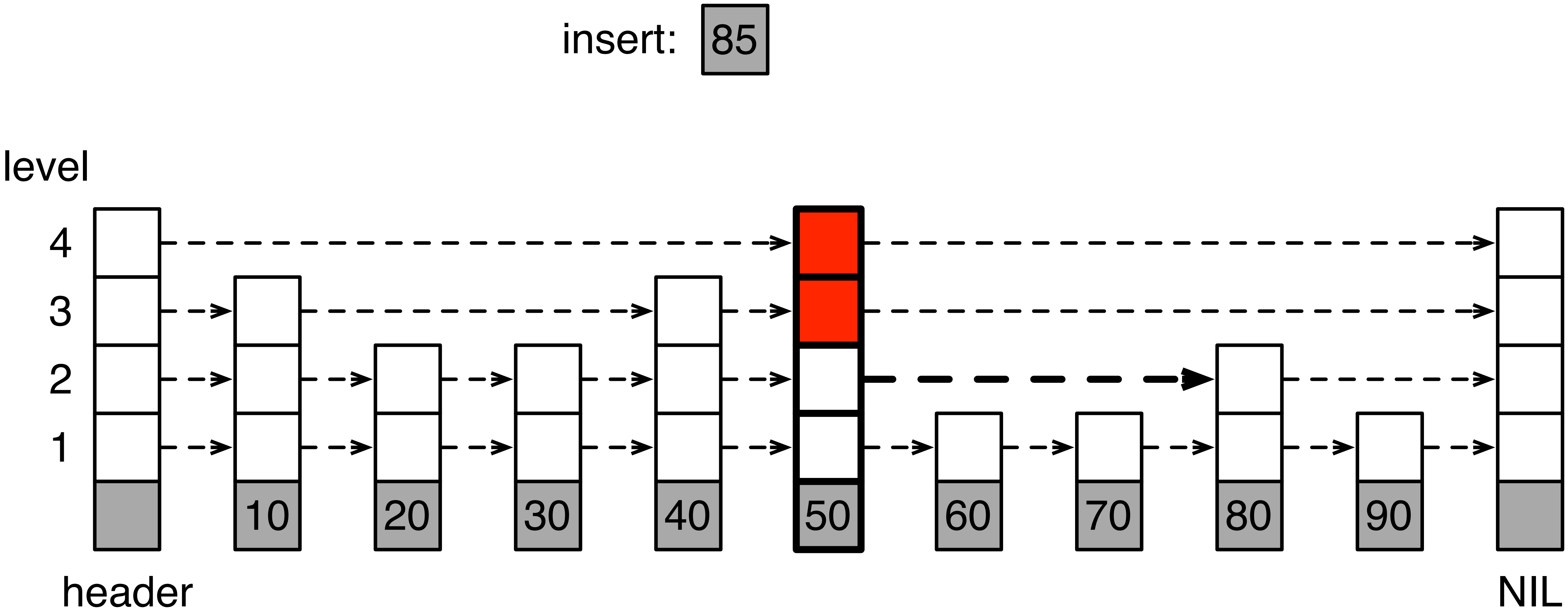


Skip List - Insertion

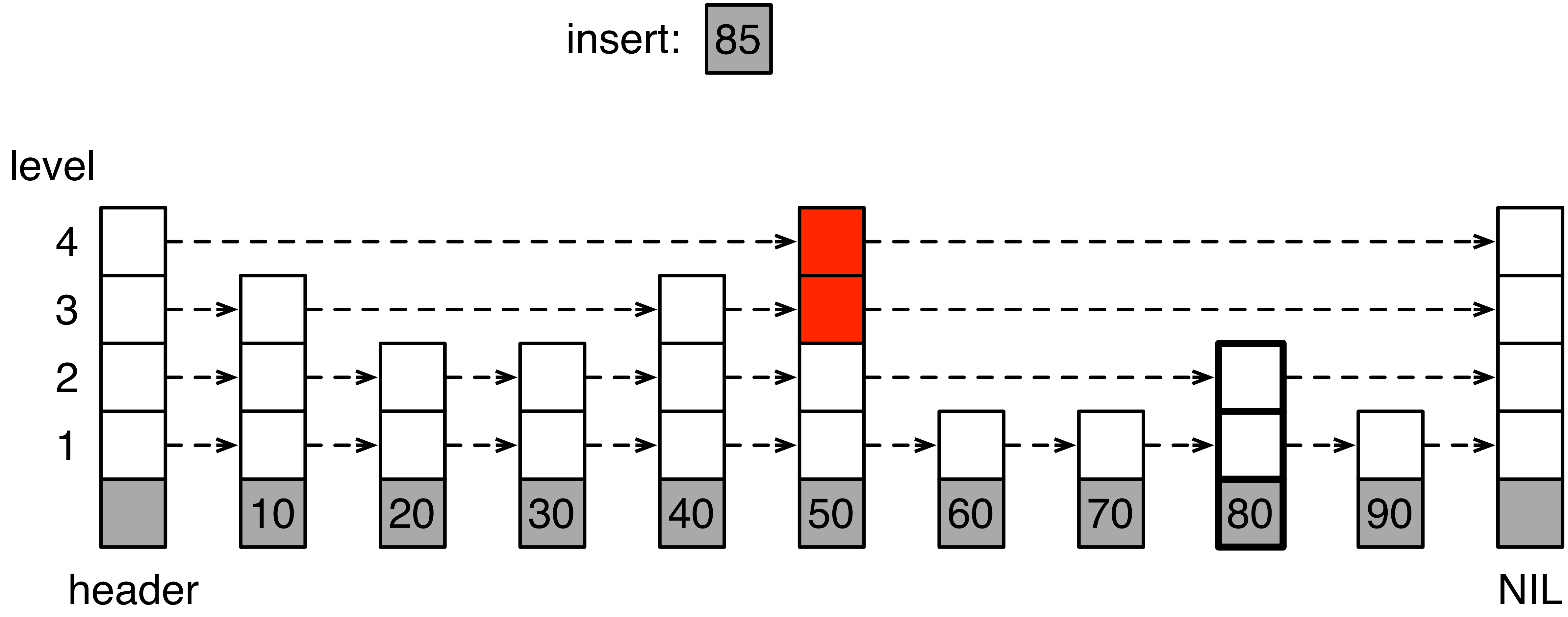
insert: 85



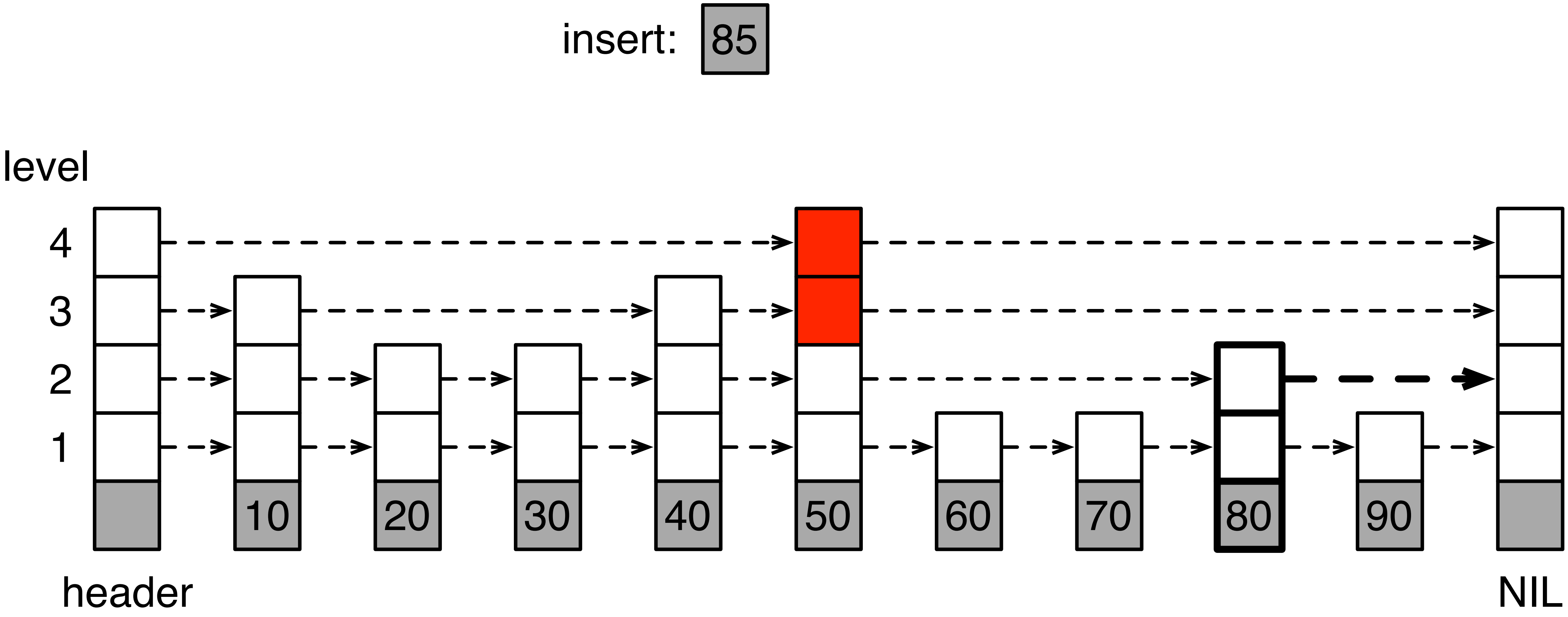
Skip List - Insertion



Skip List - Insertion

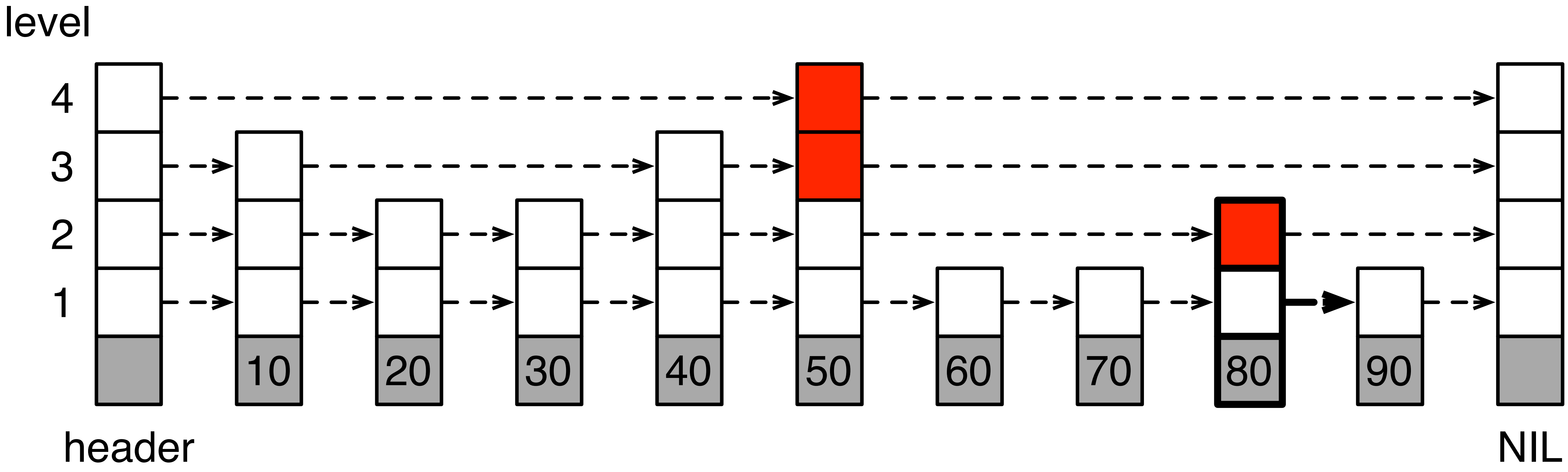


Skip List - Insertion

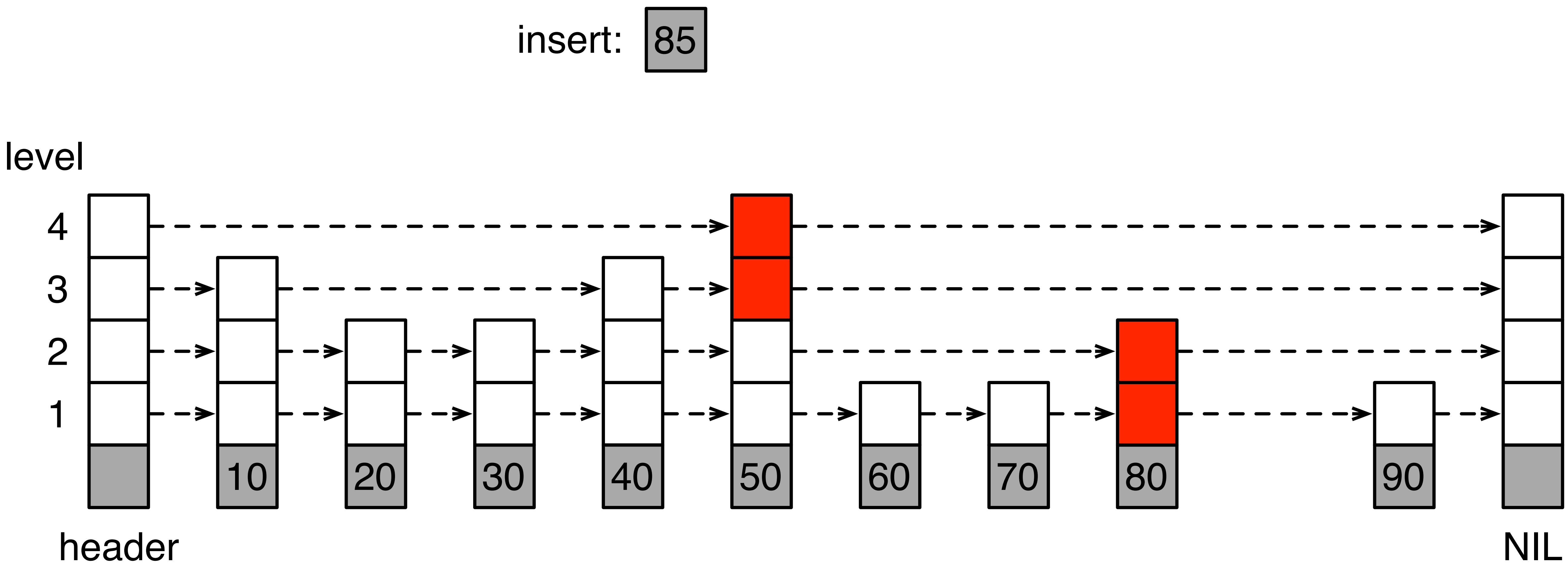


Skip List - Insertion

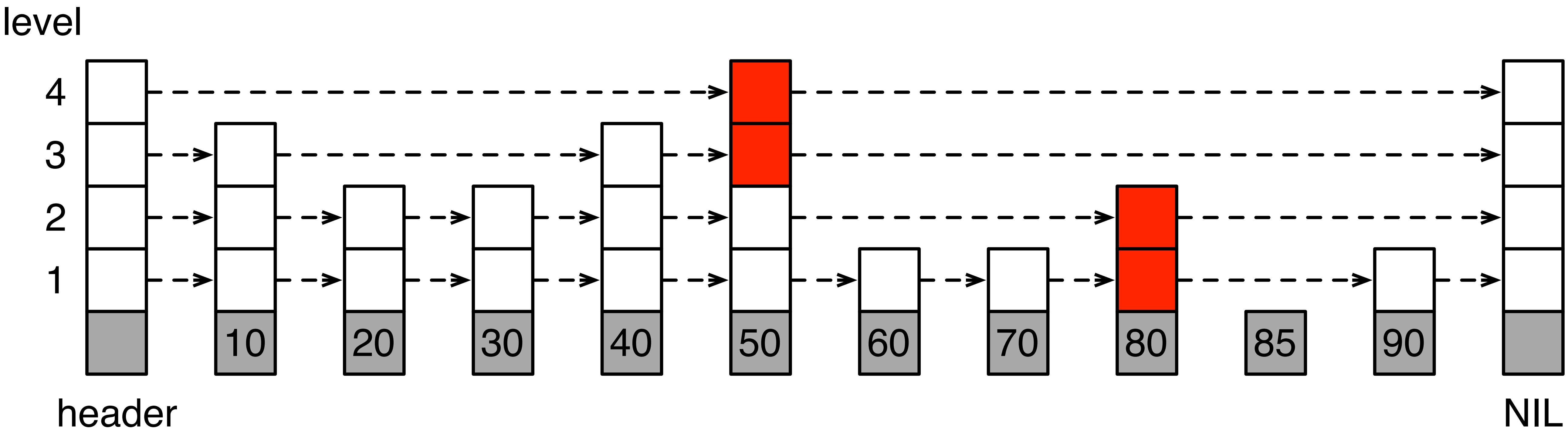
insert: 85



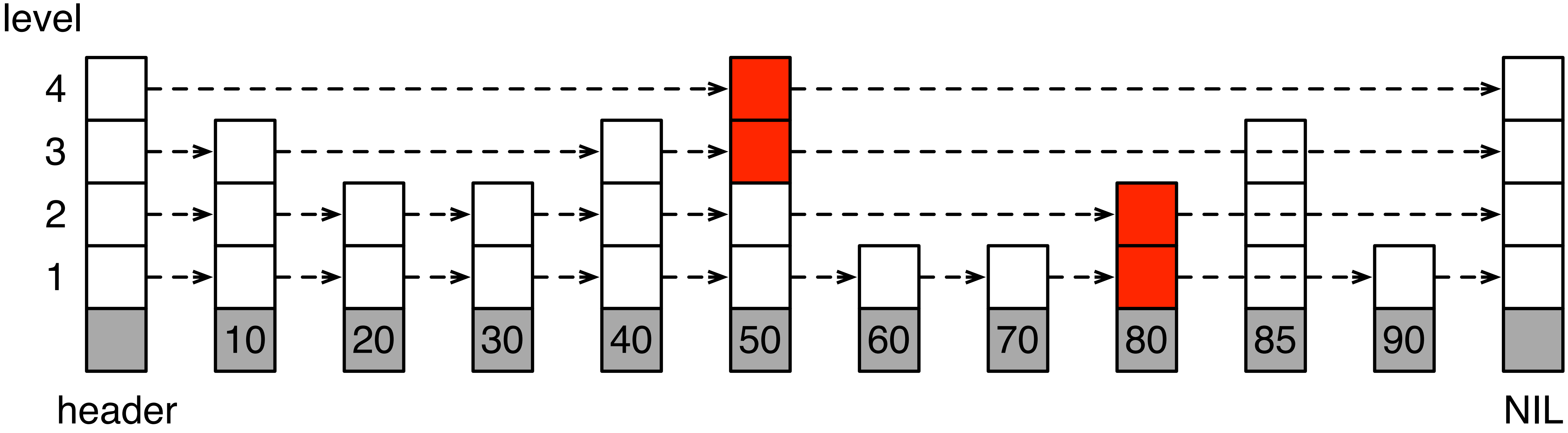
Skip List - Insertion



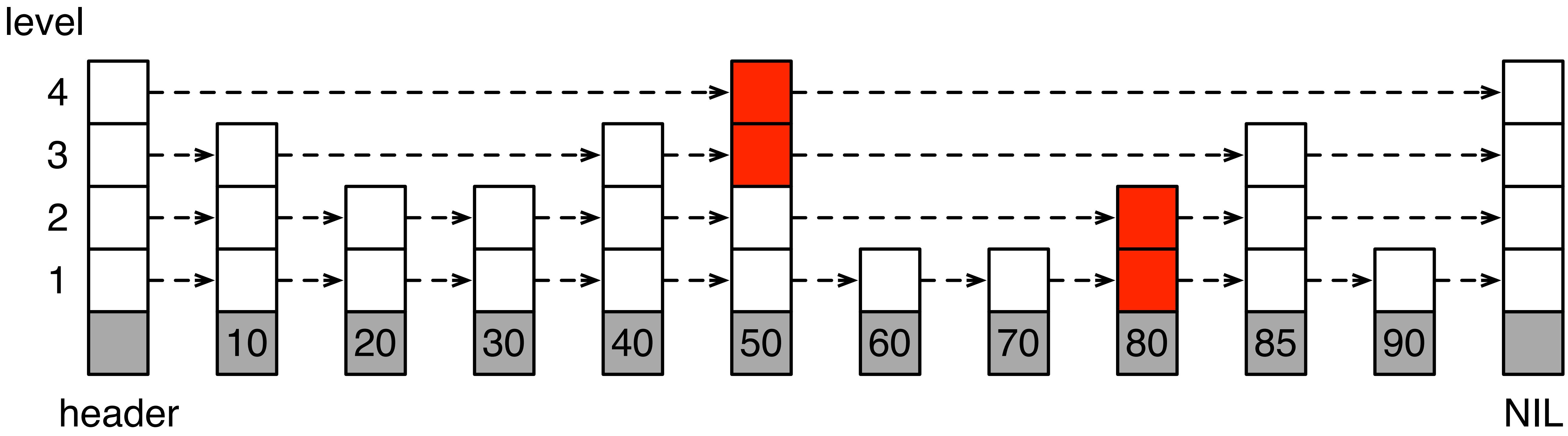
Skip List - Insertion



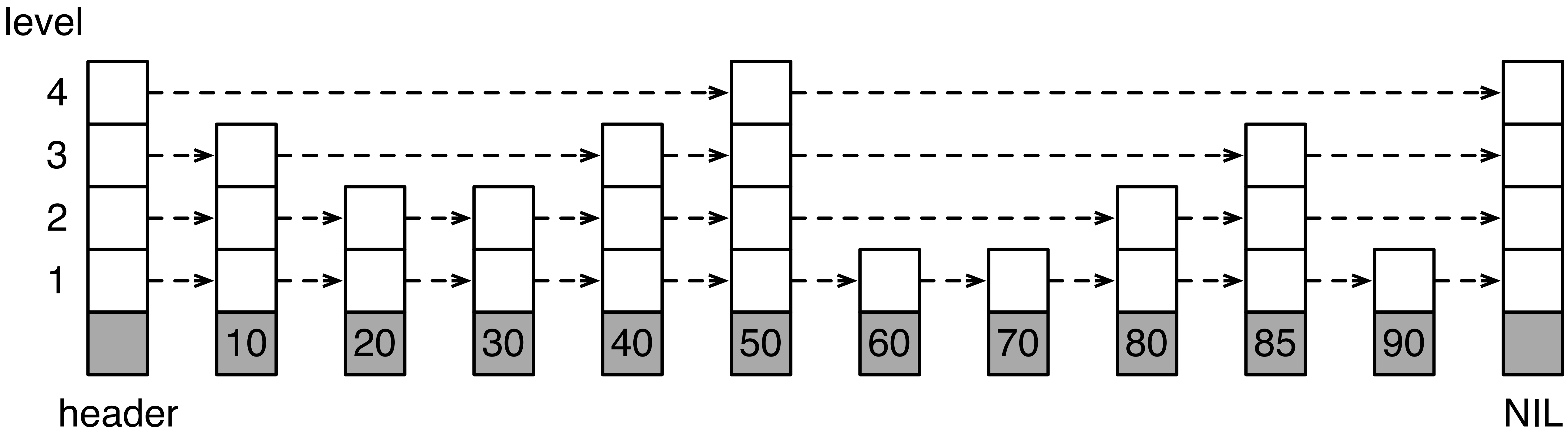
Skip List - Insertion



Skip List - Insertion



Skip List - Insertion



Skip Lists

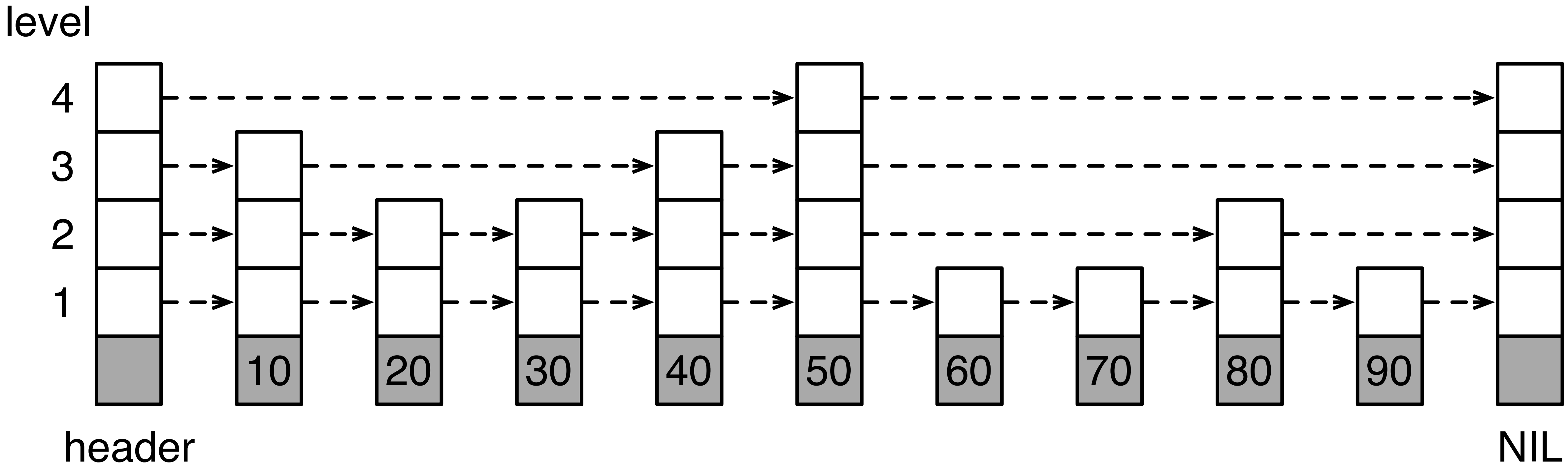
ADVANTAGES

- order of insertion does not matter
- easy to implement
- constant factor performance advantage over trees
- space efficient

DISADVANTAGES

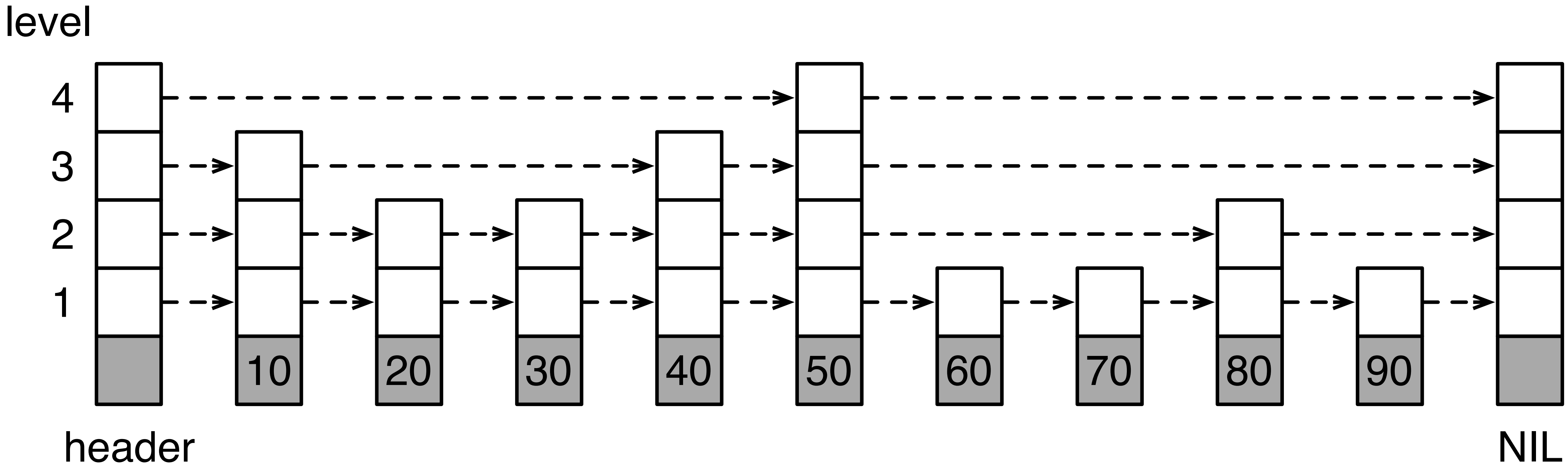
- cannot be paginated efficiently
- insertion/deletion requires global search

skip list

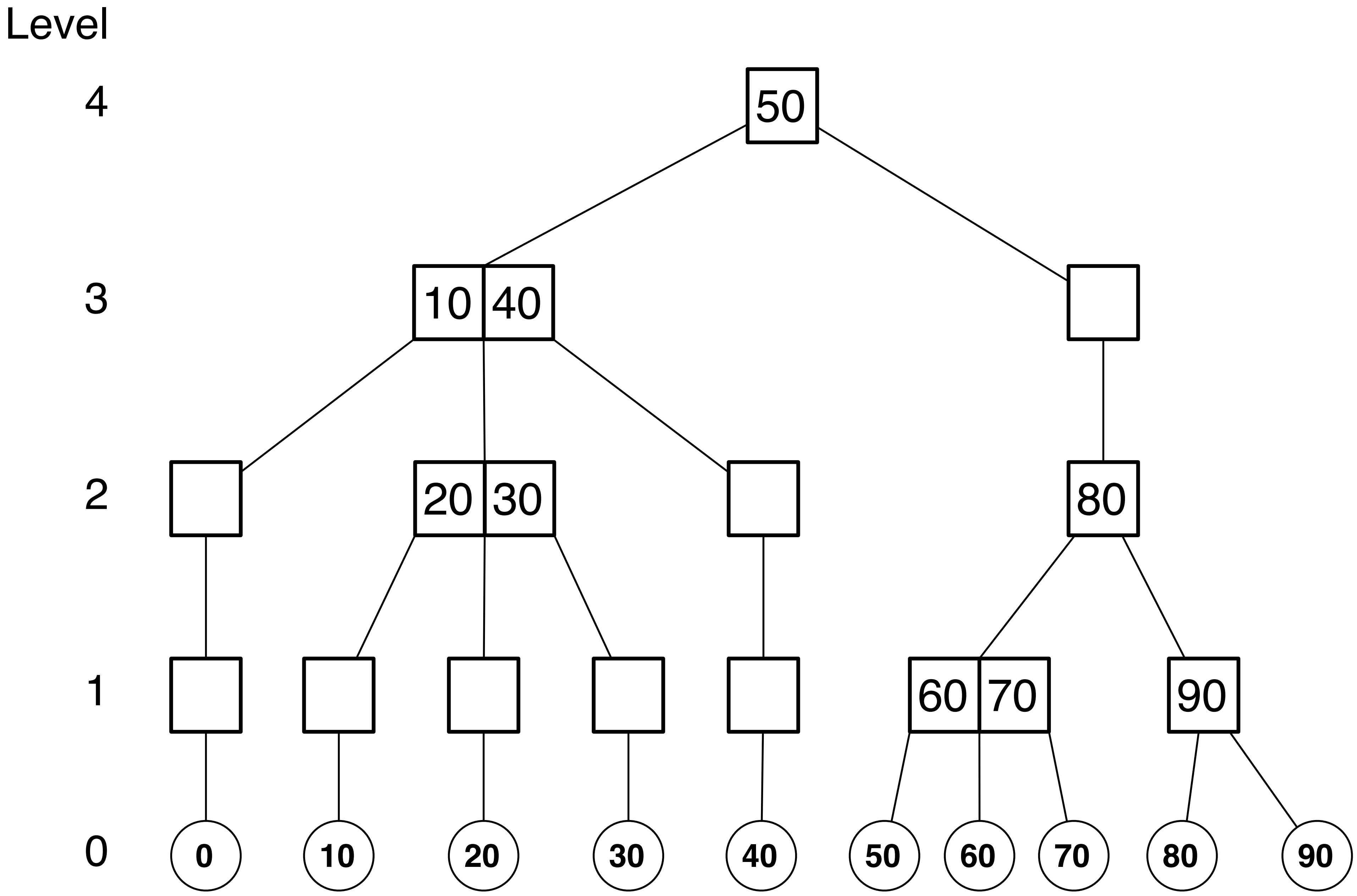


skip lists can be re-envisioned as trees
(Xavier Messeguer, 1997)

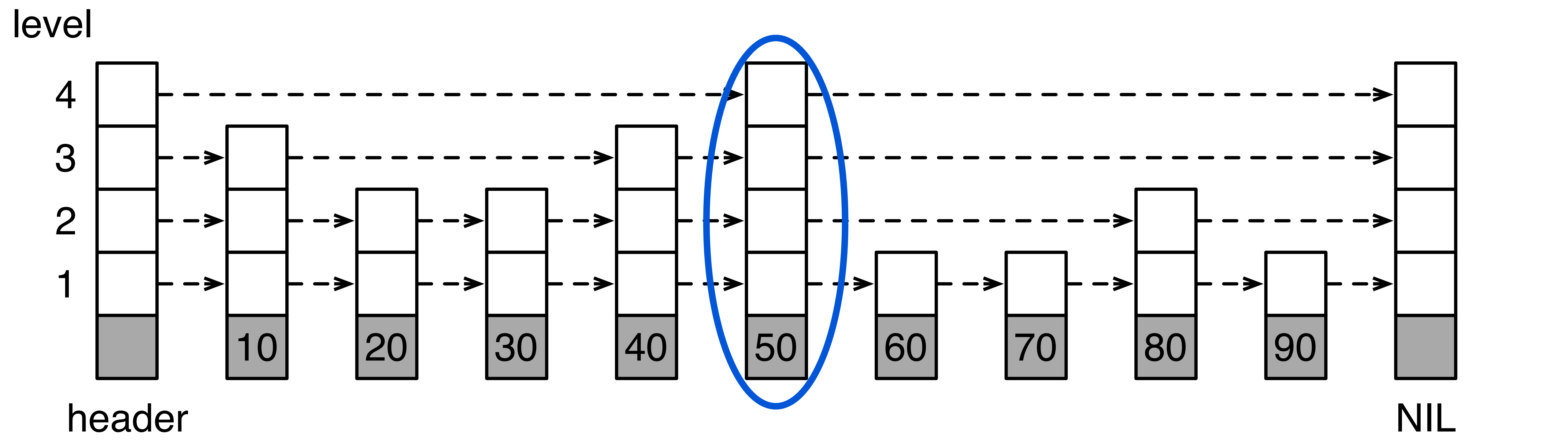
skip list



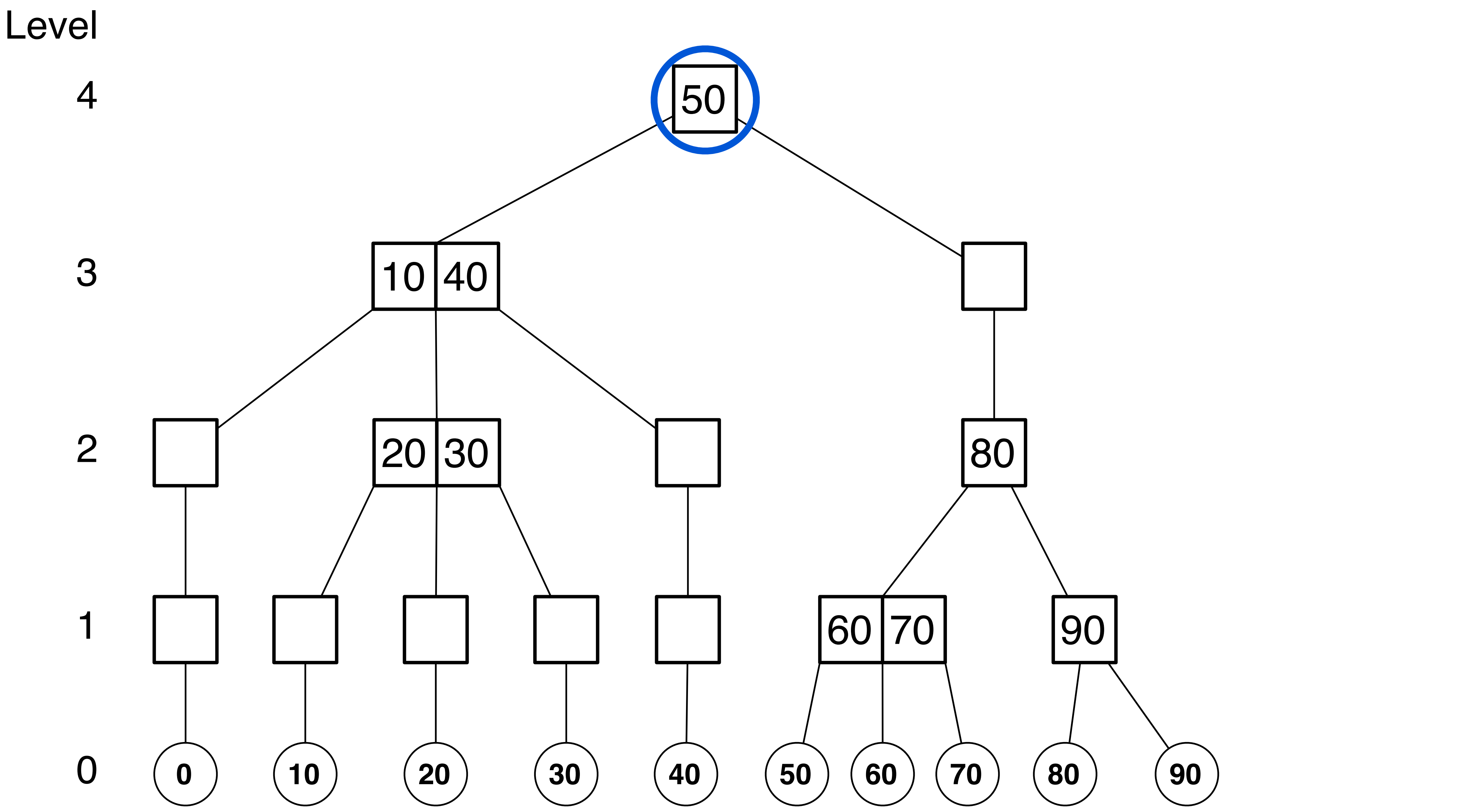
skip tree



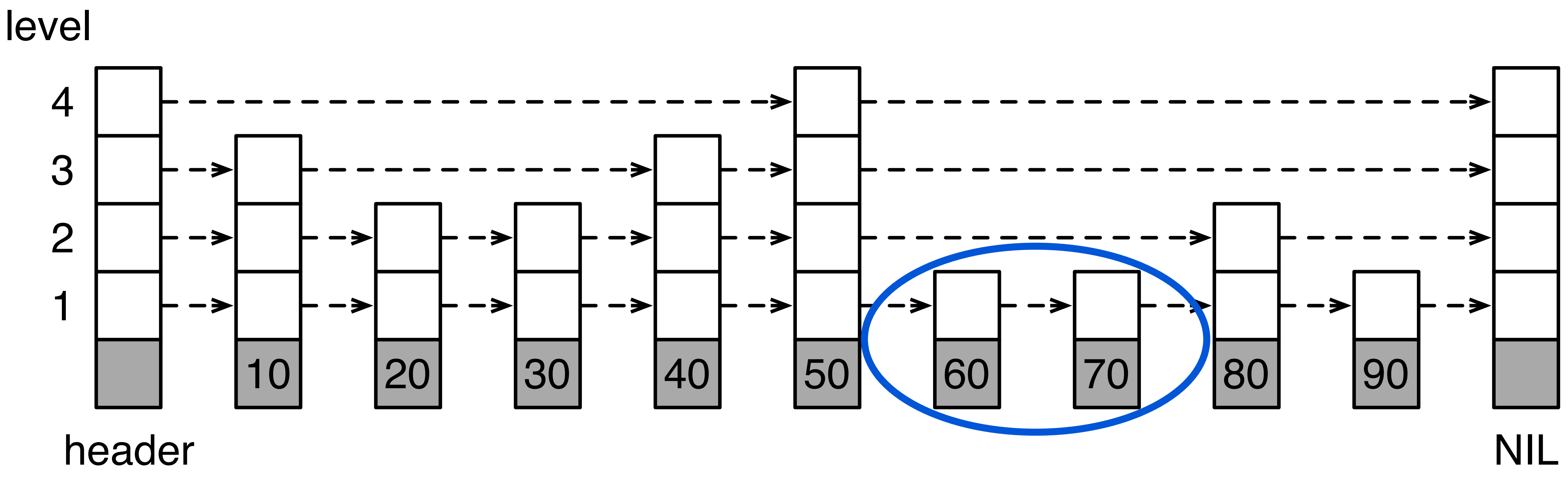
skip list



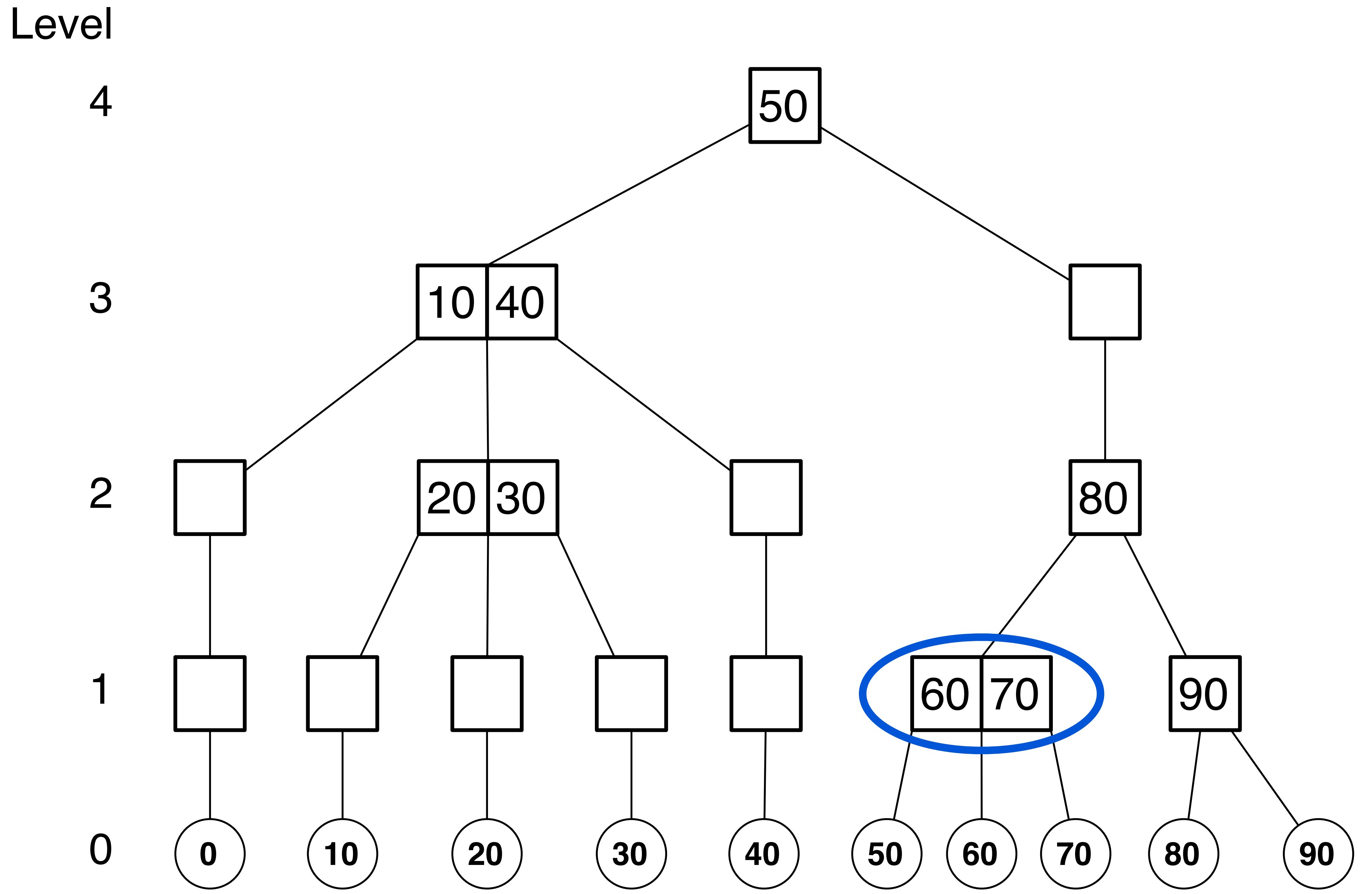
skip tree



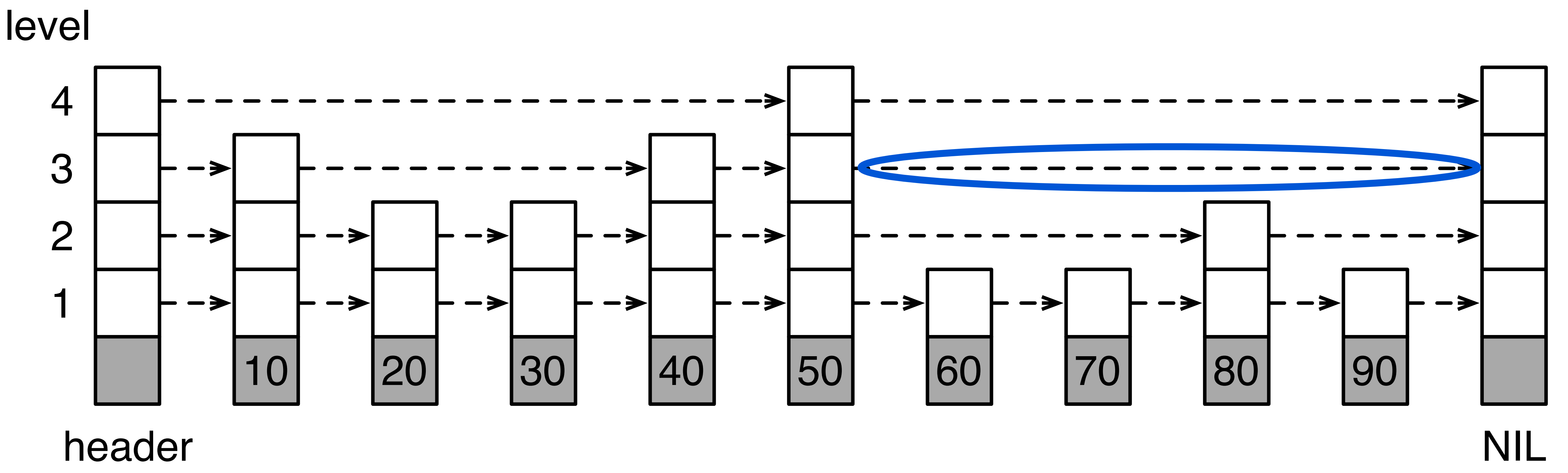
skip list



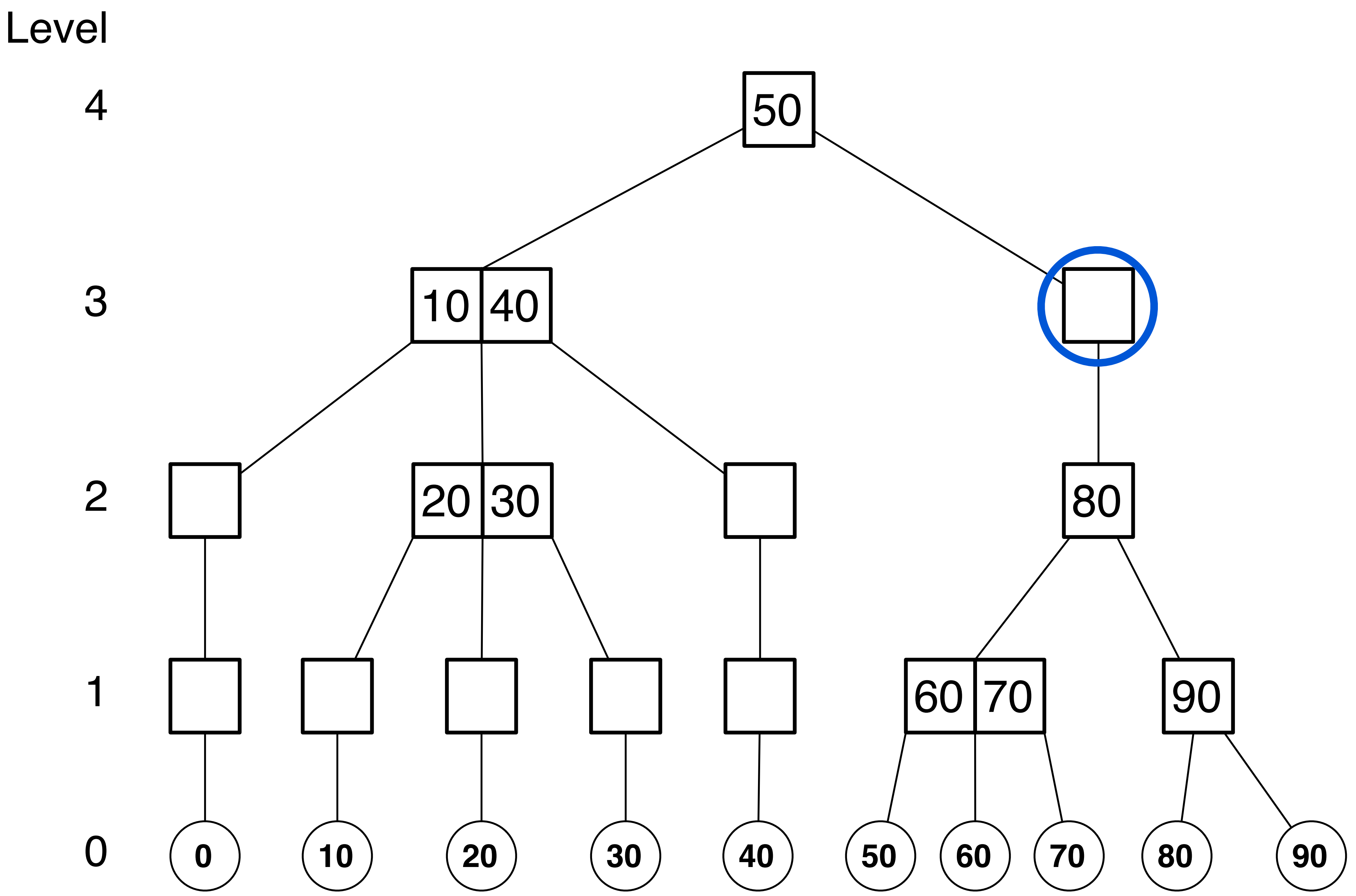
skip tree



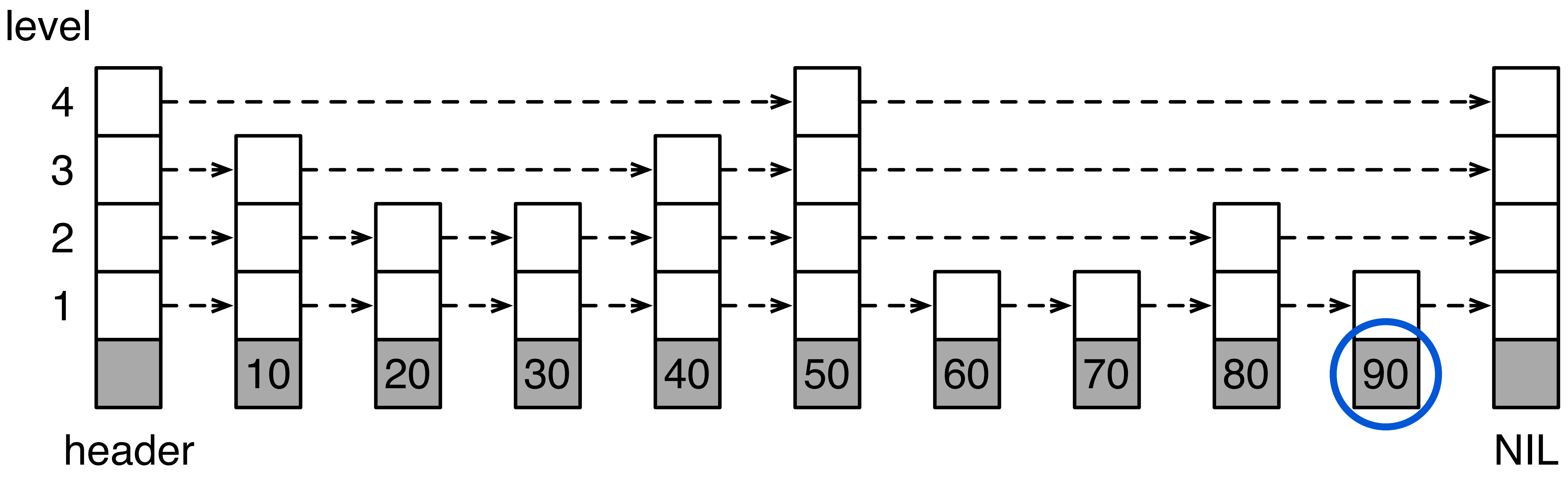
skip list



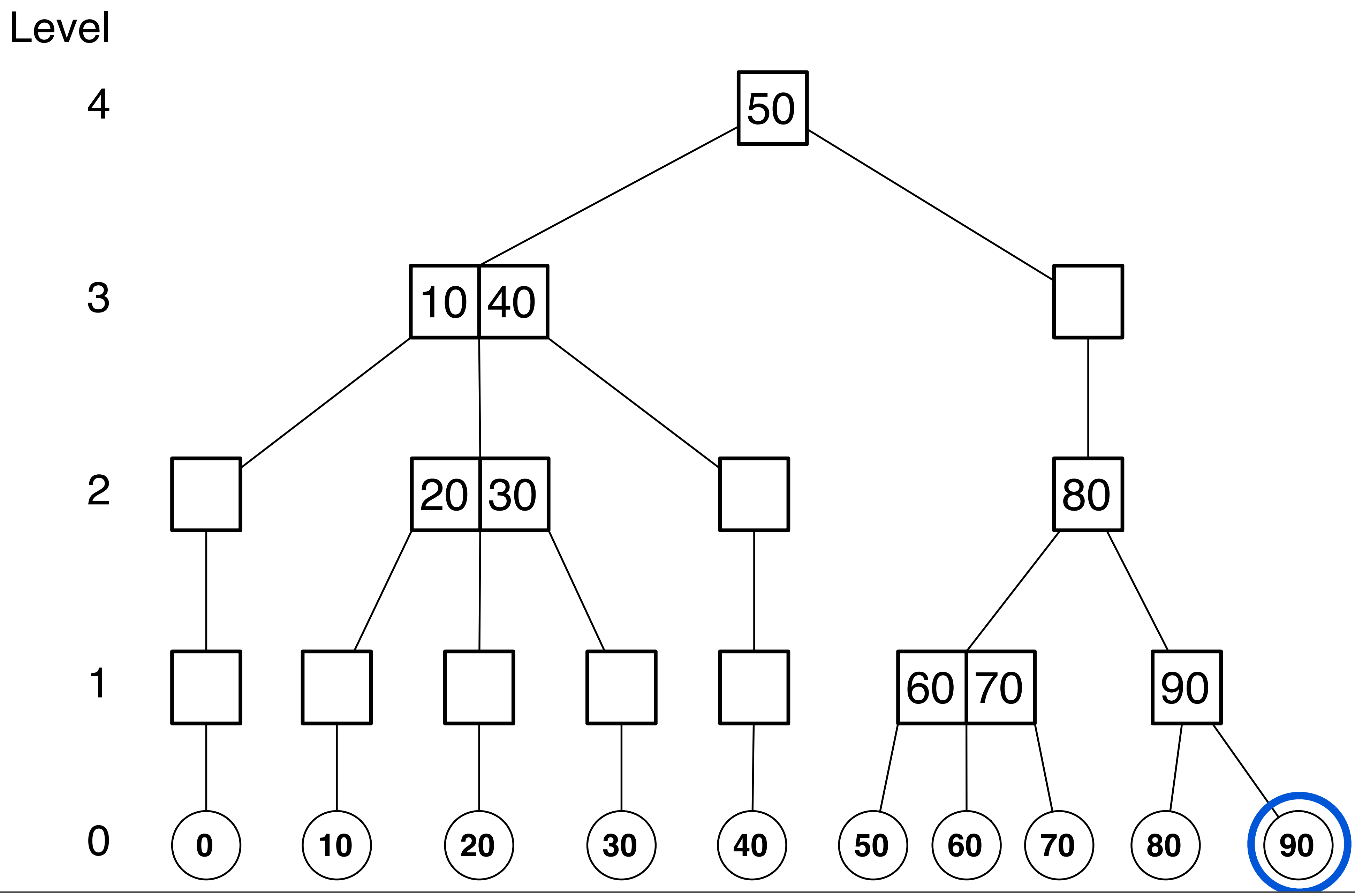
skip tree



skip list

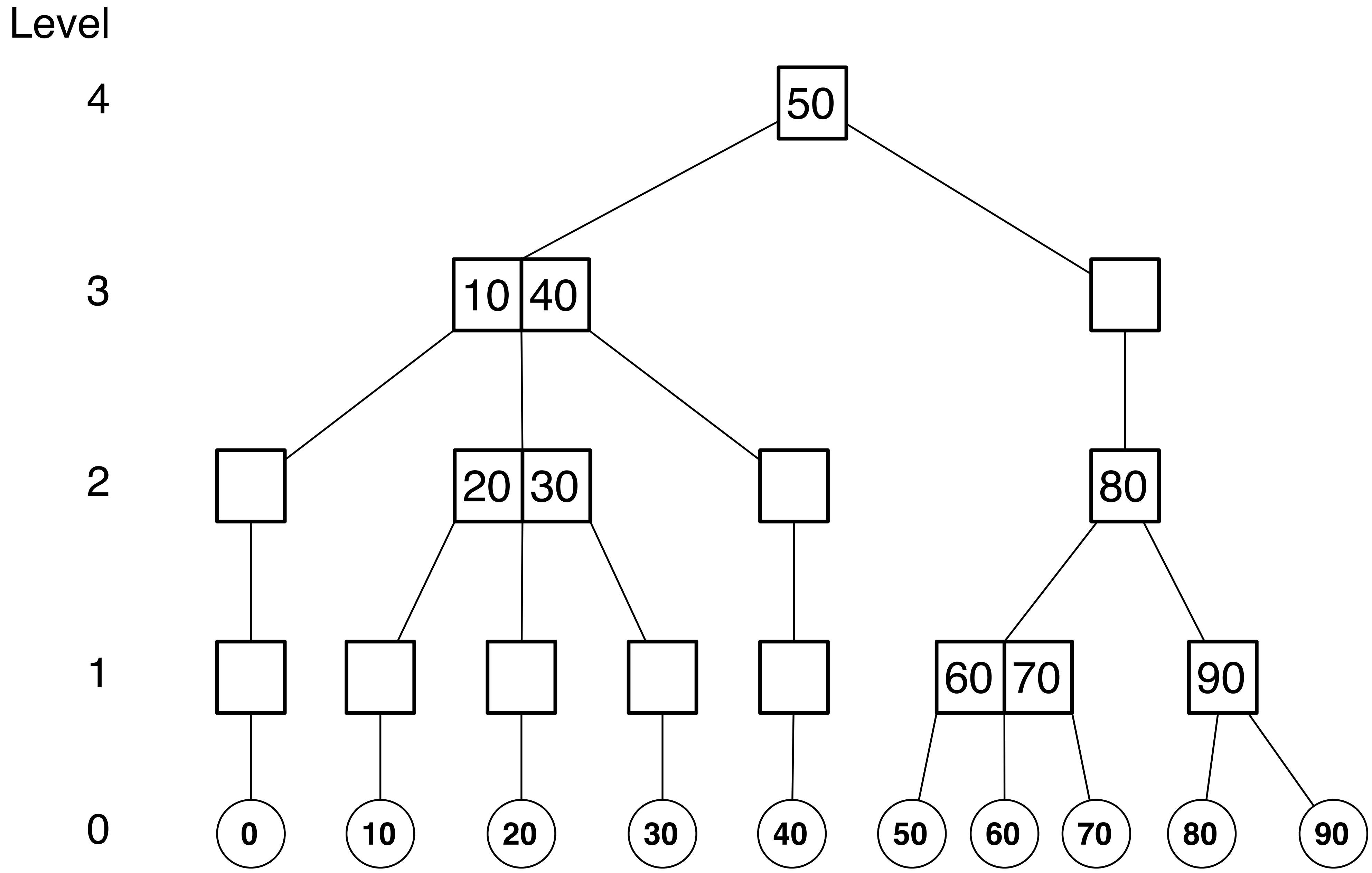


skip tree



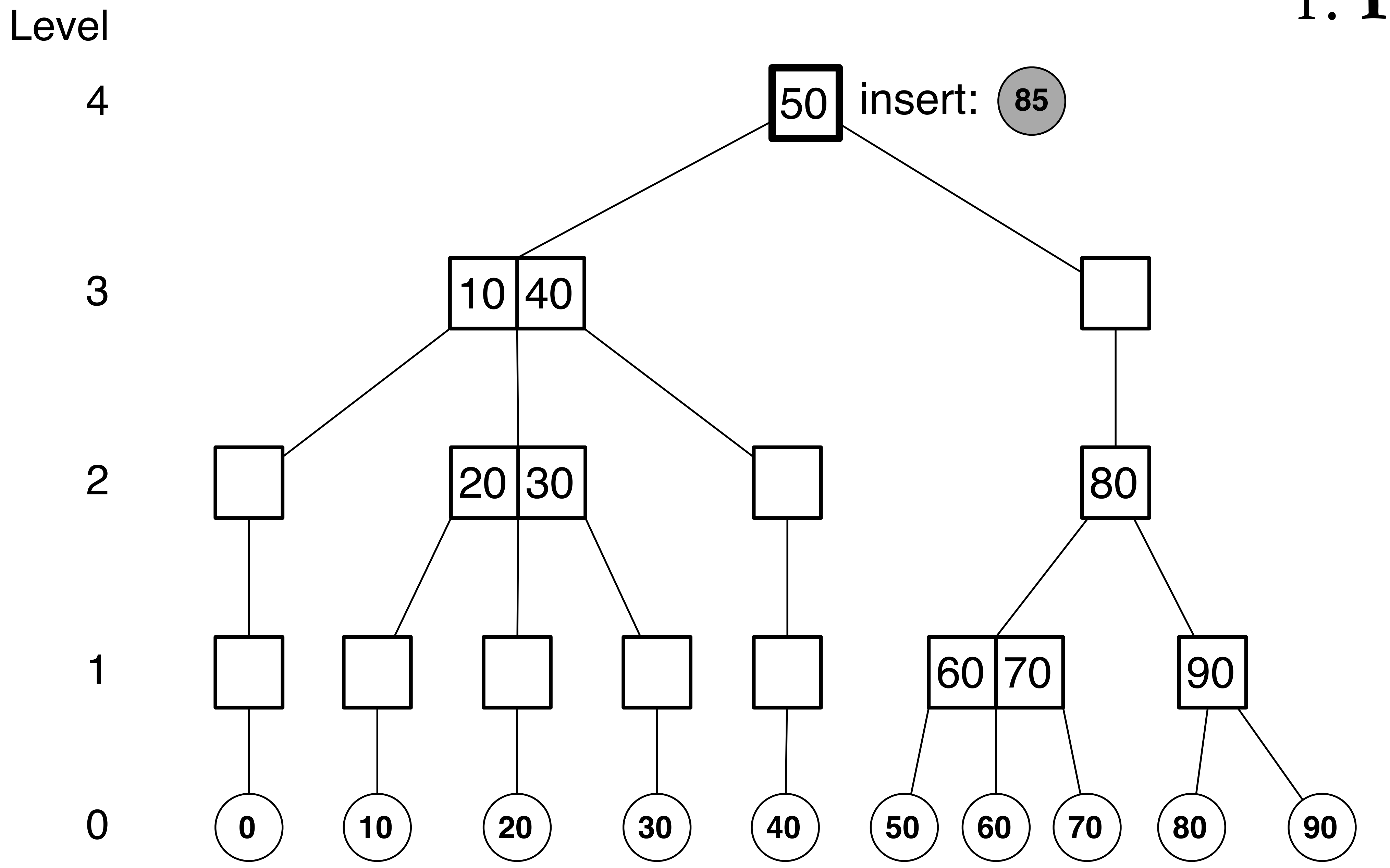
Skip Trees - Insertion

insert: 85



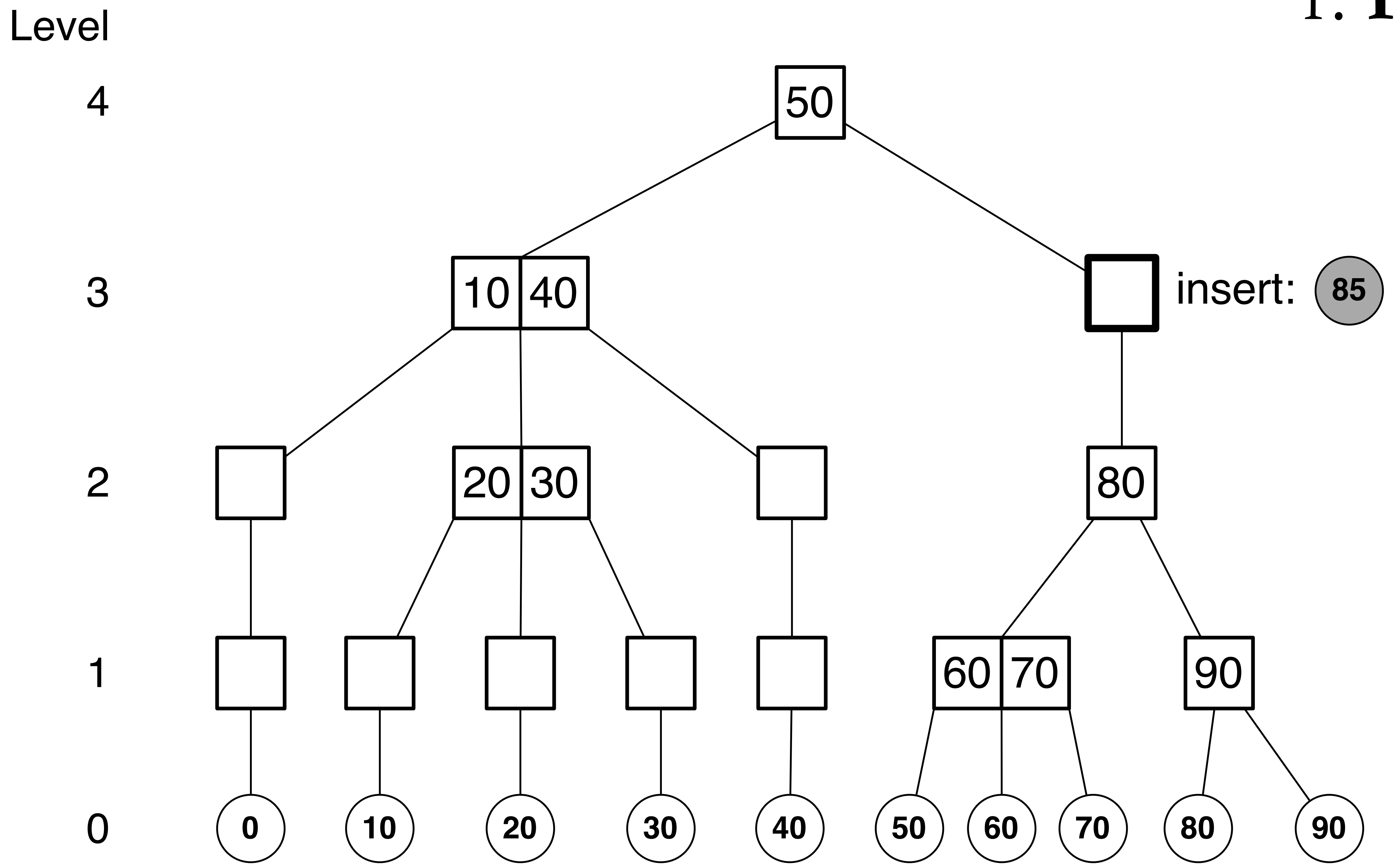
Skip Trees - Insertion

1. **Percolate**(1)



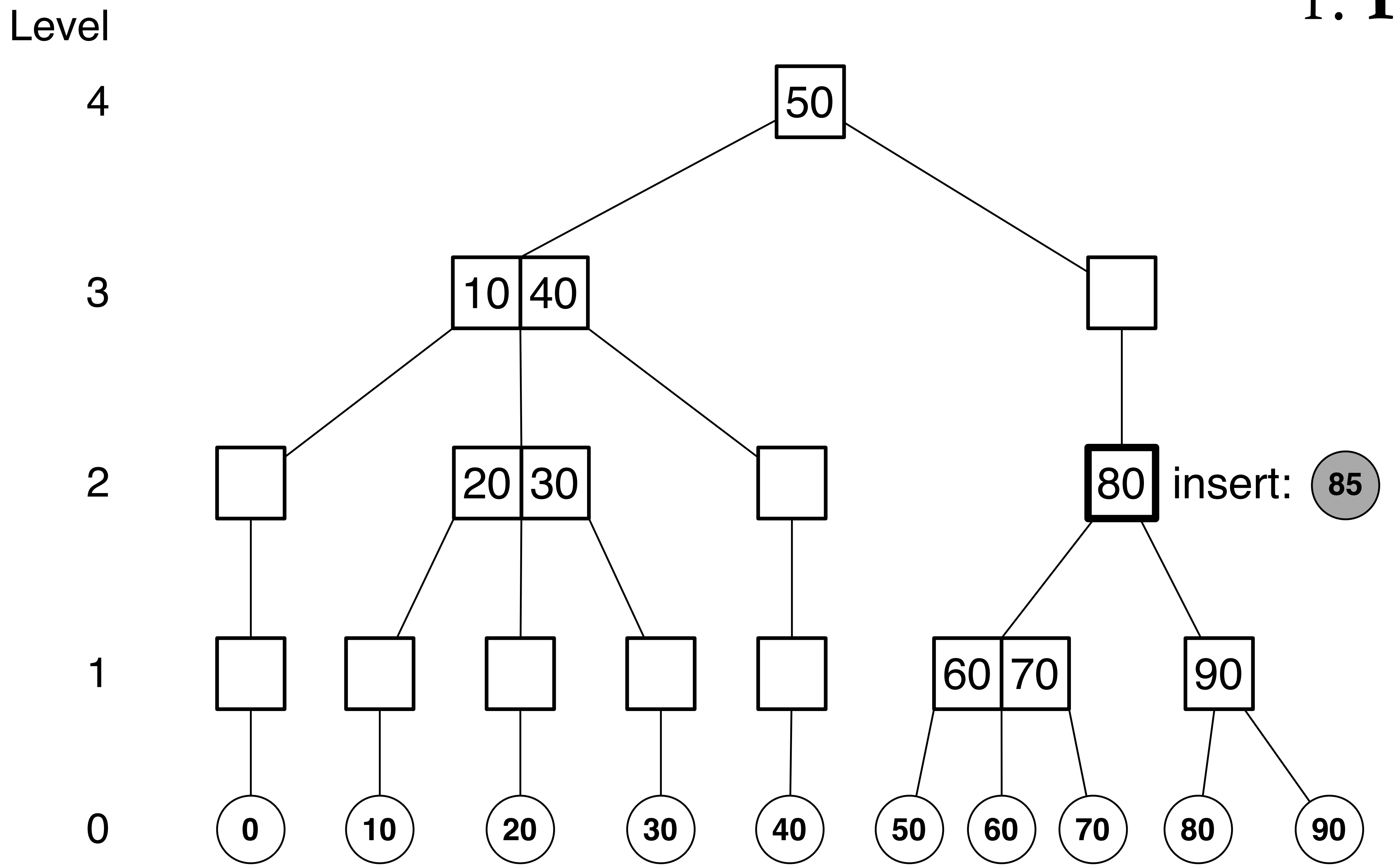
Skip Trees - Insertion

1. **Percolate**(2)



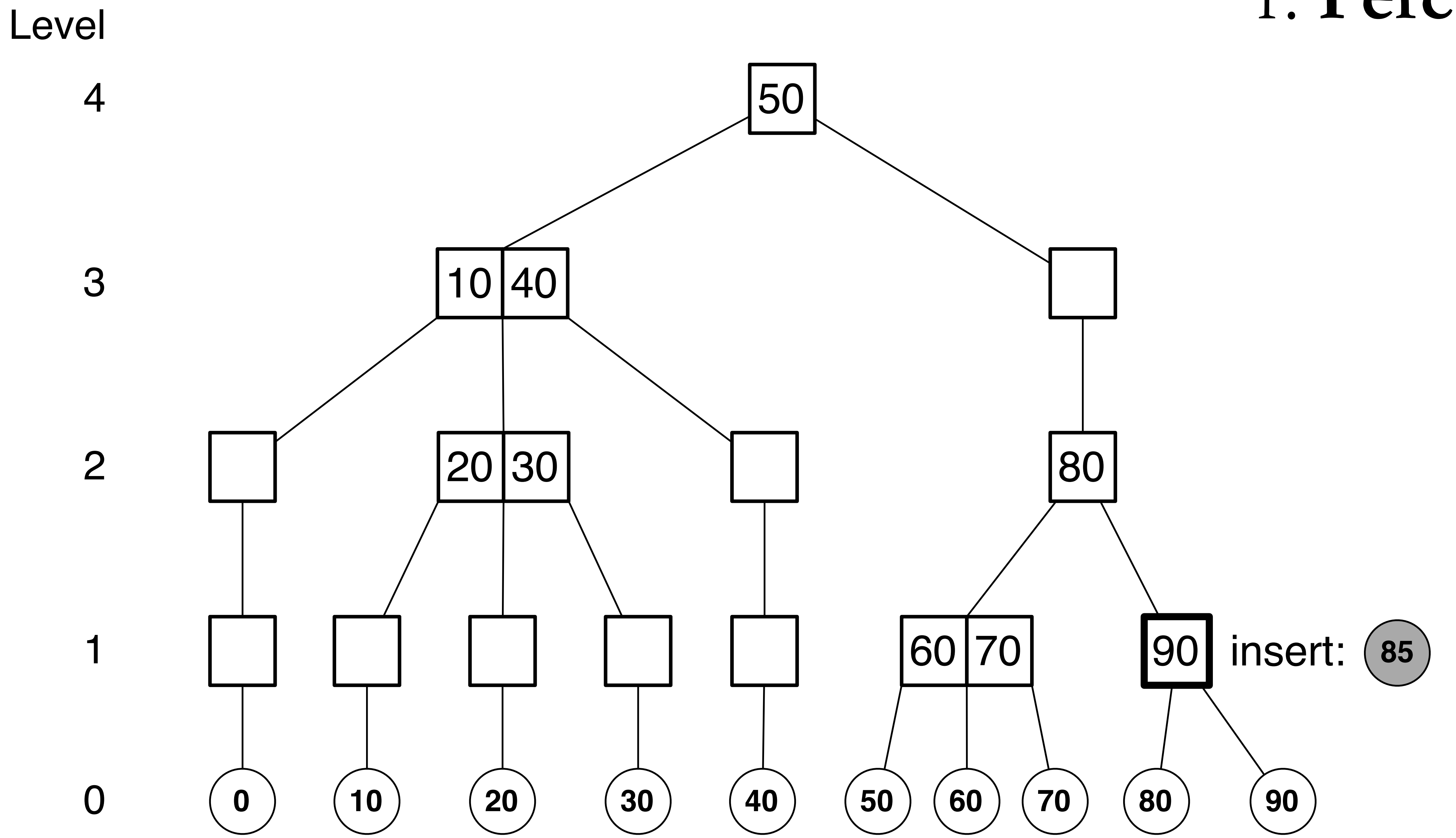
Skip Trees - Insertion

1. **Percolate**(3)



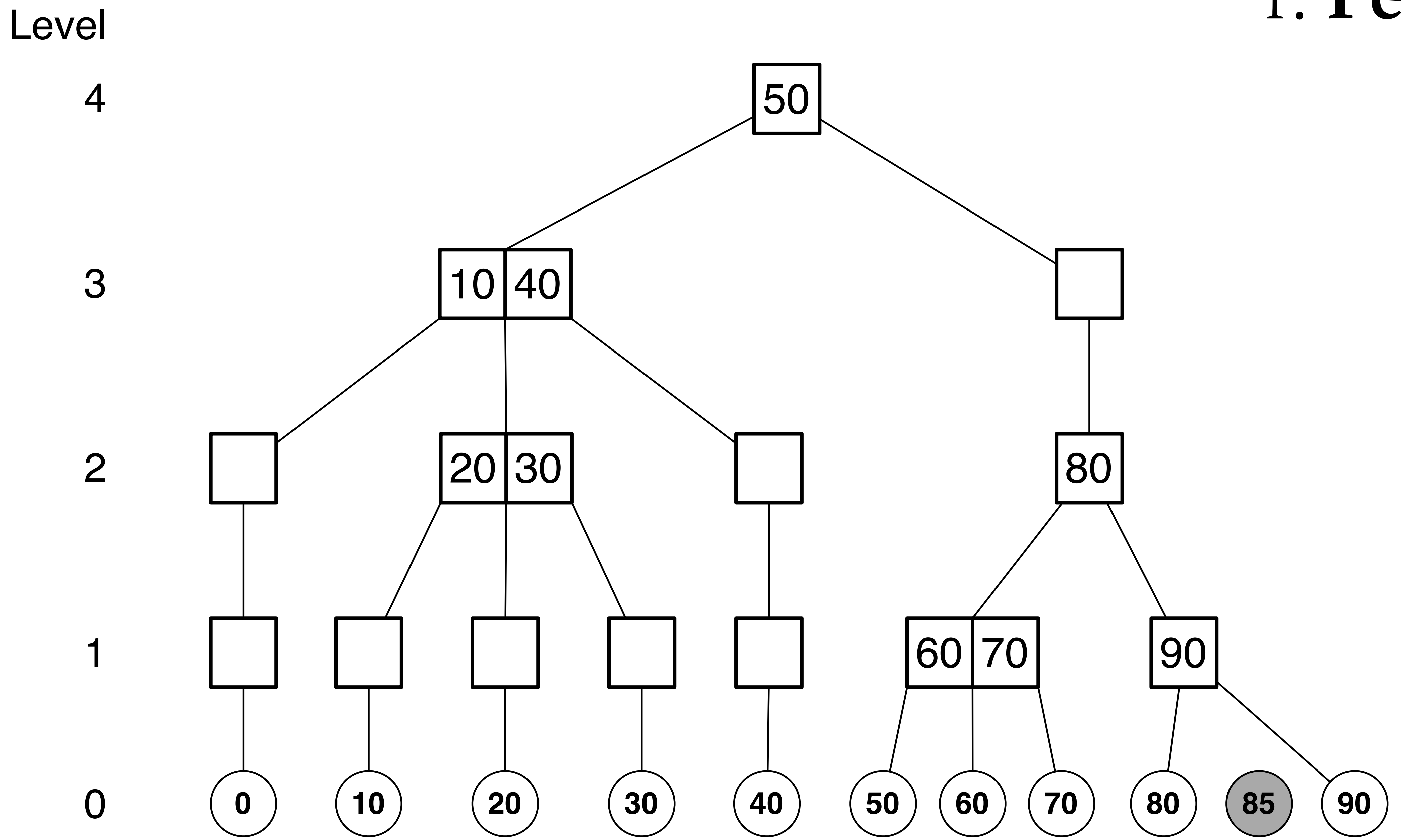
Skip Trees - Insertion

1. **Percolate**(4)



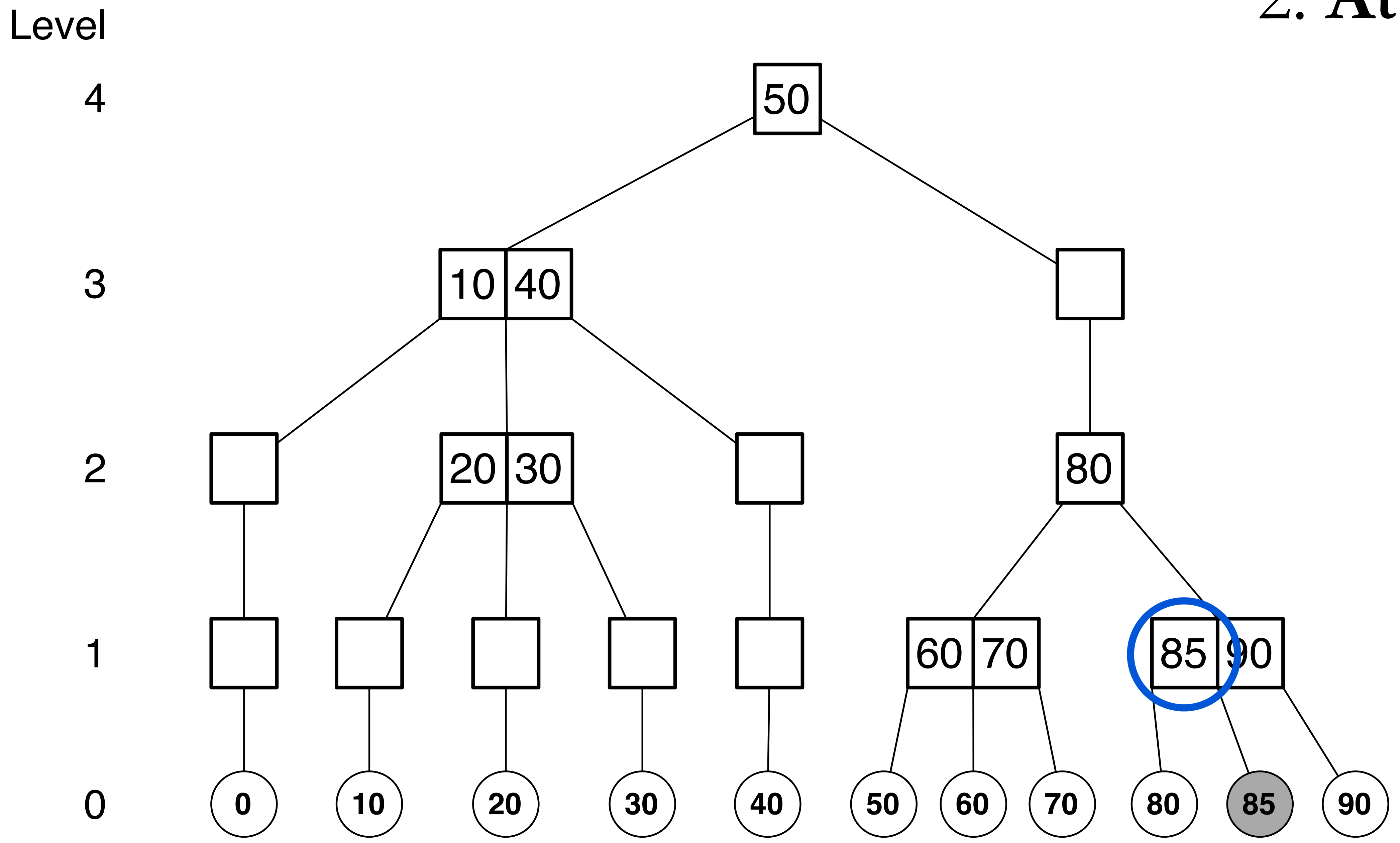
Skip Trees - Insertion

1. **Percolate**(5)

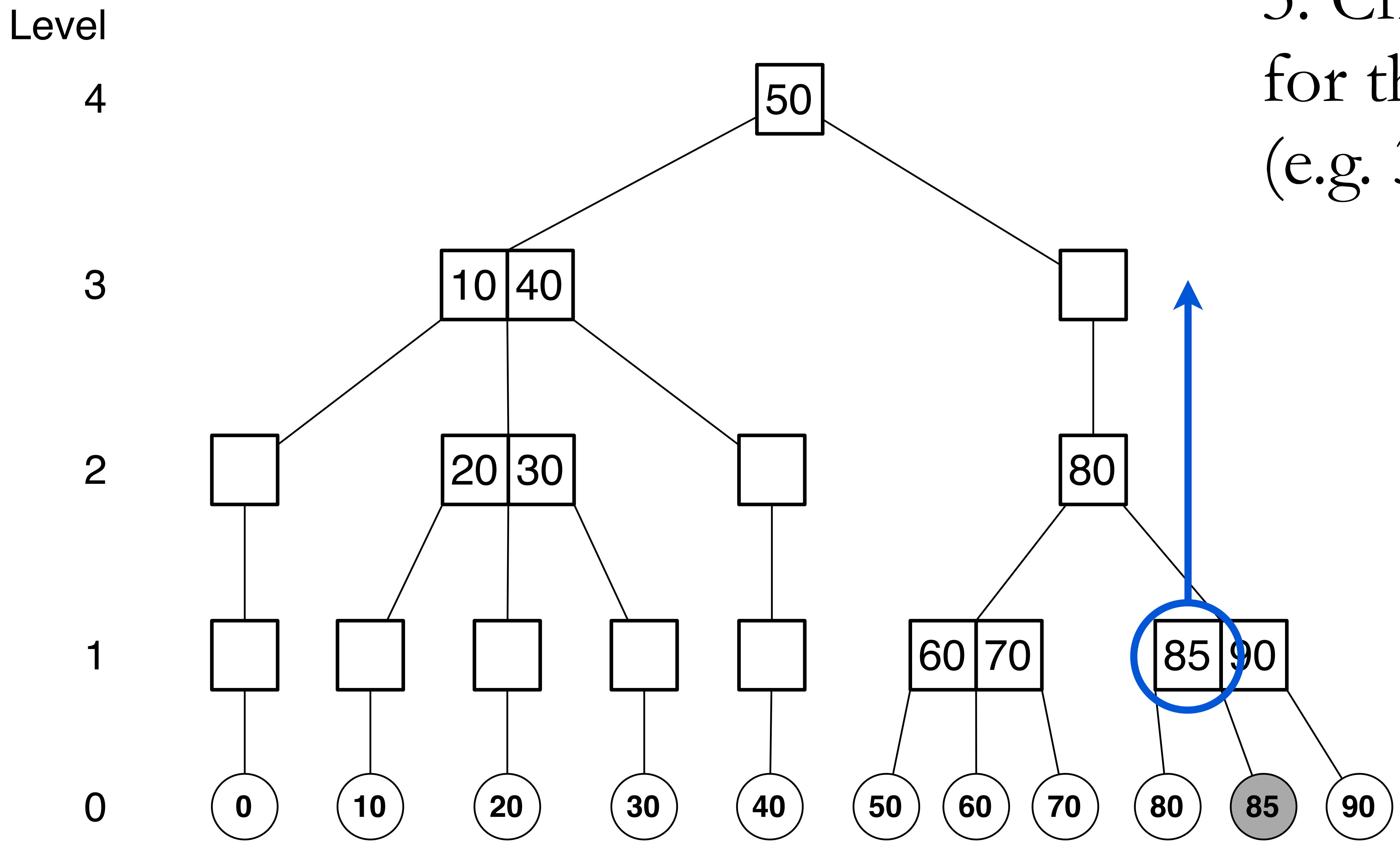


Skip Trees - Insertion

2. **Attach**



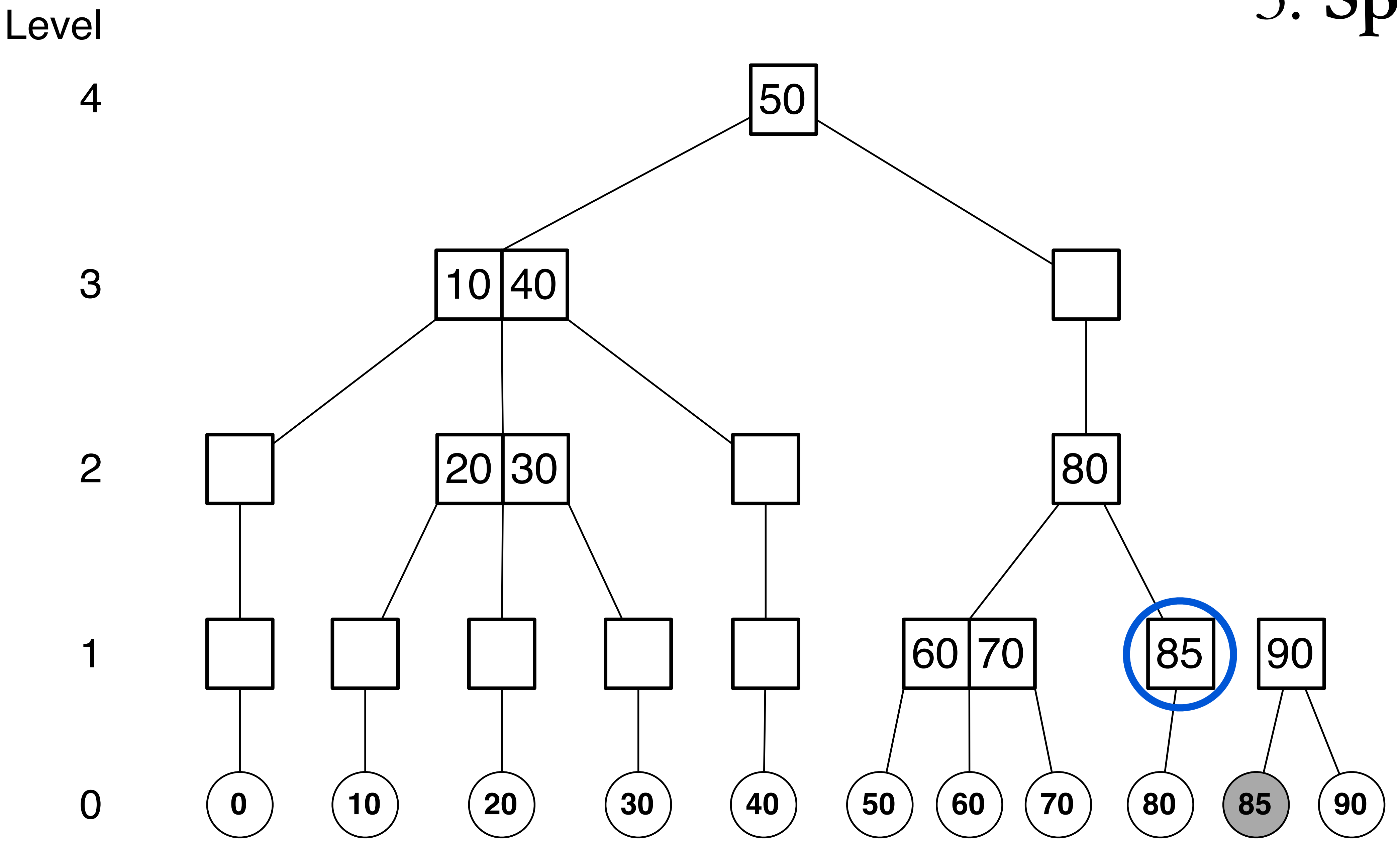
Skip Trees - Insertion



3. Choose a level for the new key (e.g. 3)

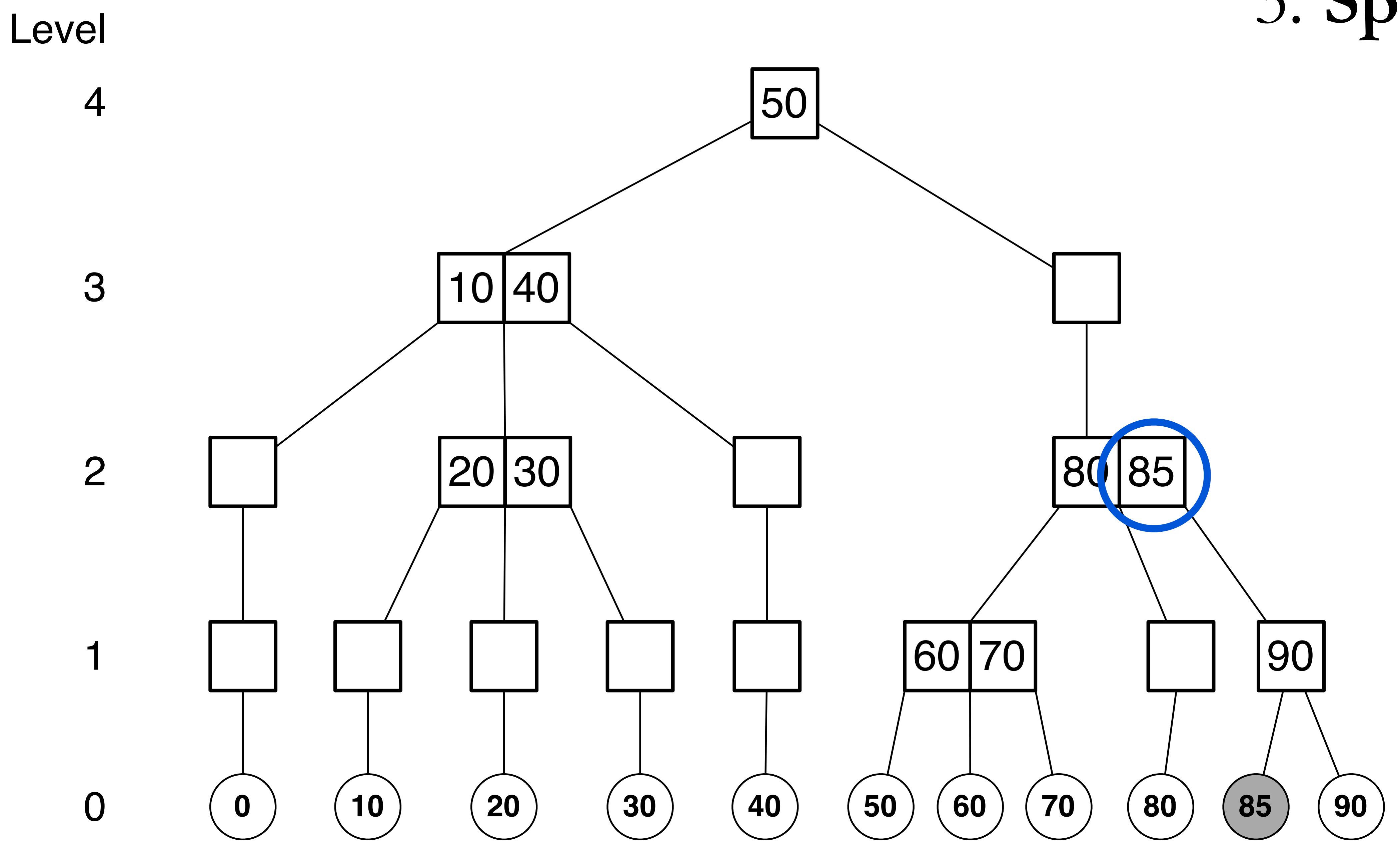
Skip Trees - Insertion

3. **Split**(1)



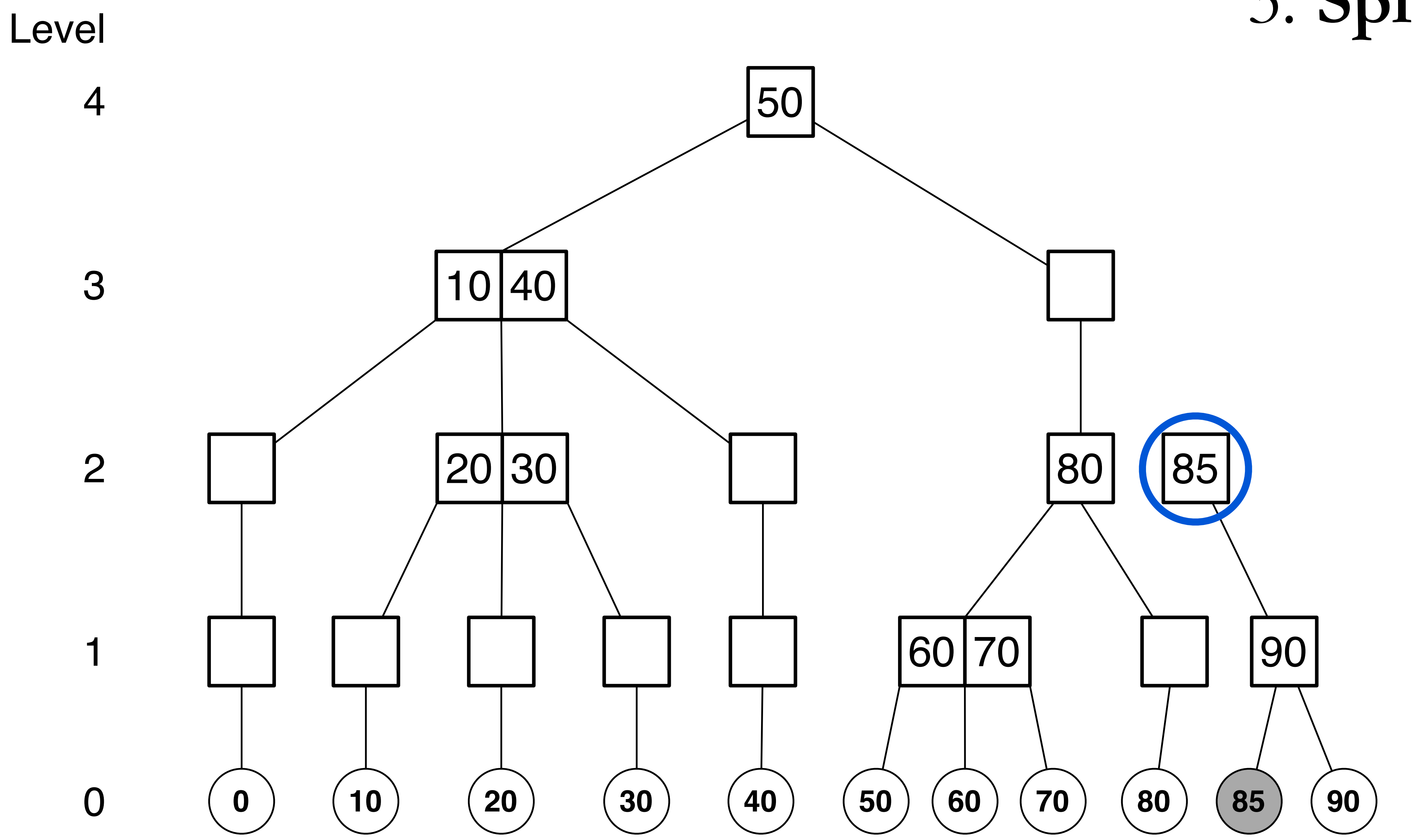
Skip Trees - Insertion

3. **Split**(1)



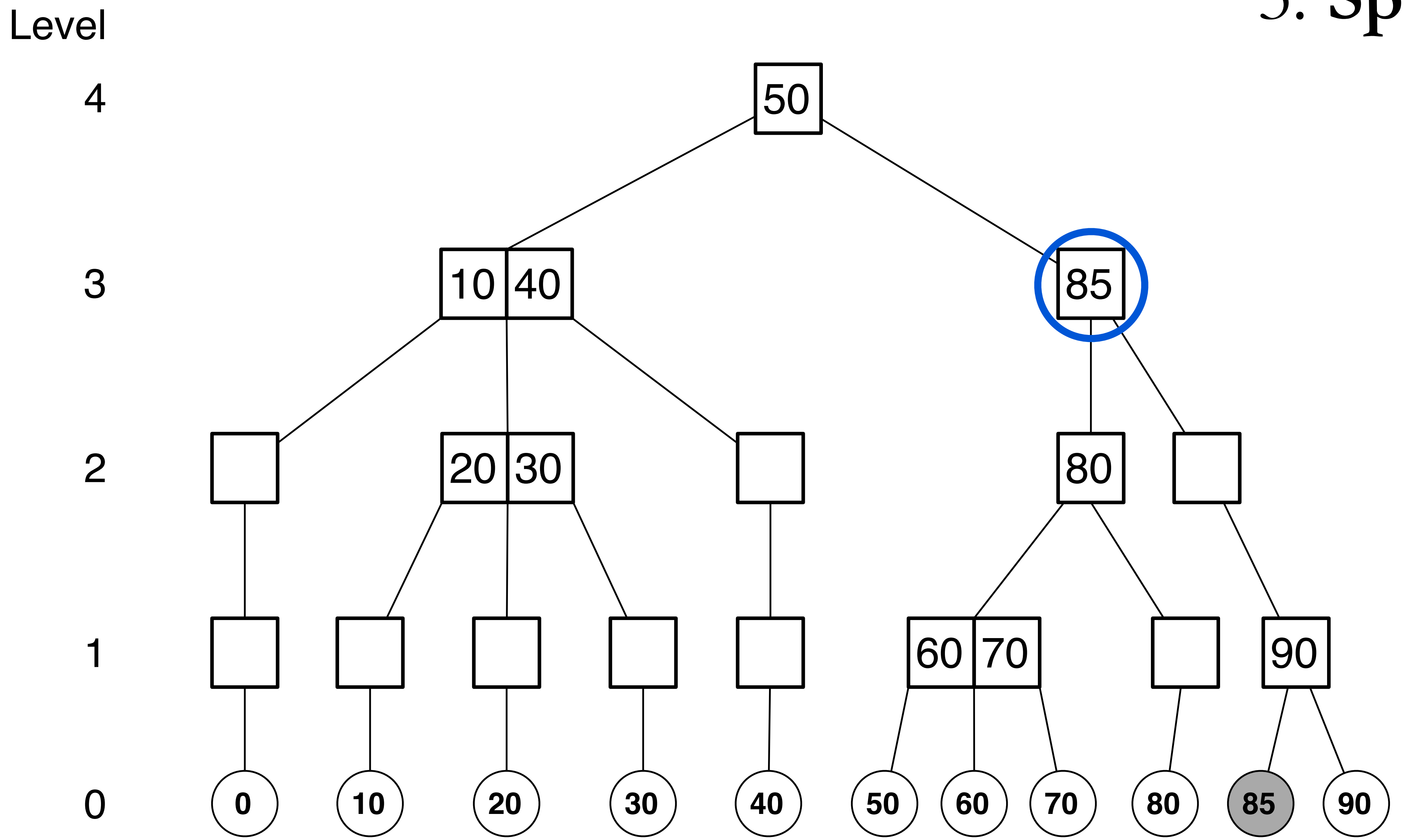
Skip Trees - Insertion

3. **Split**(2)

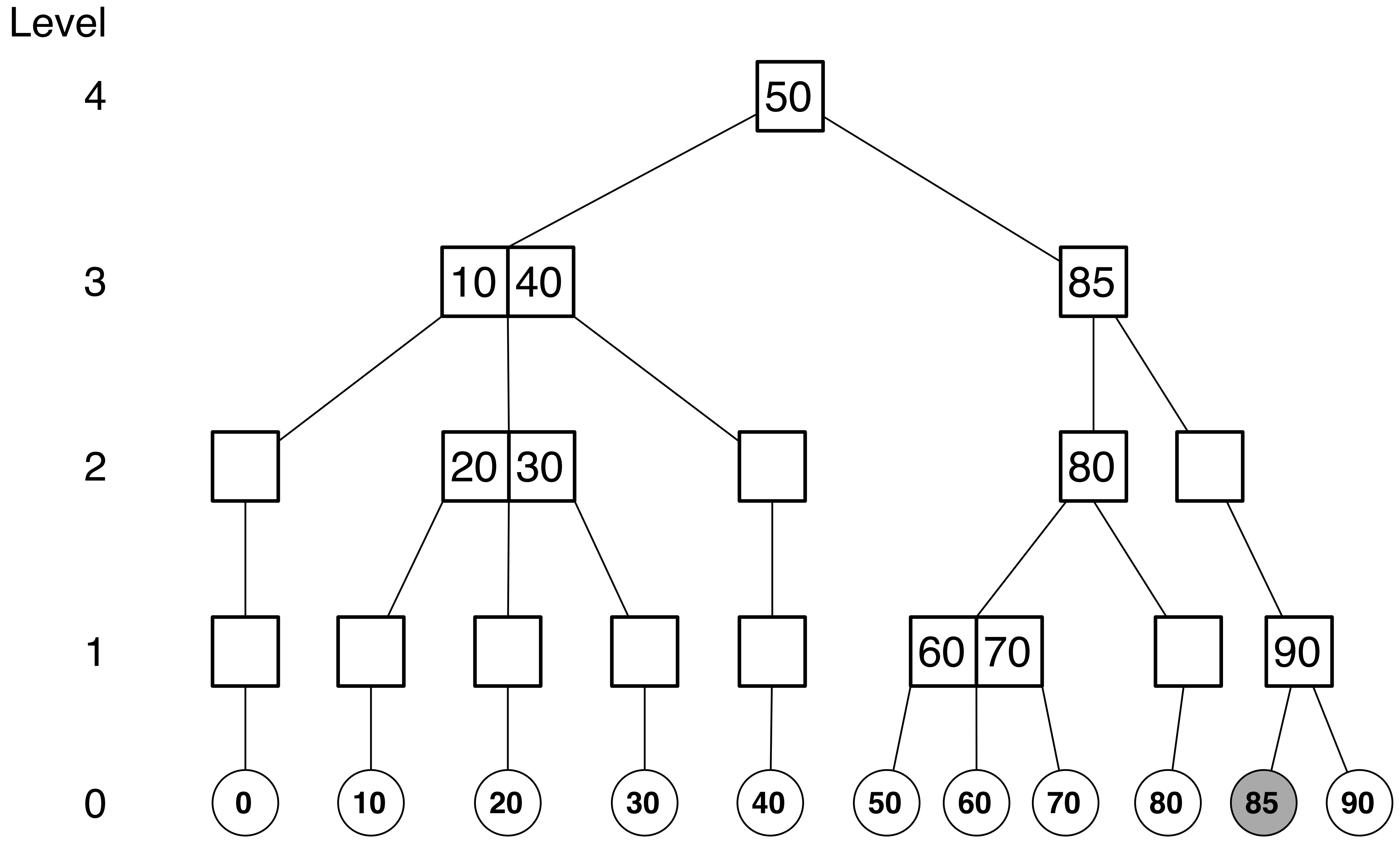


Skip Trees - Insertion

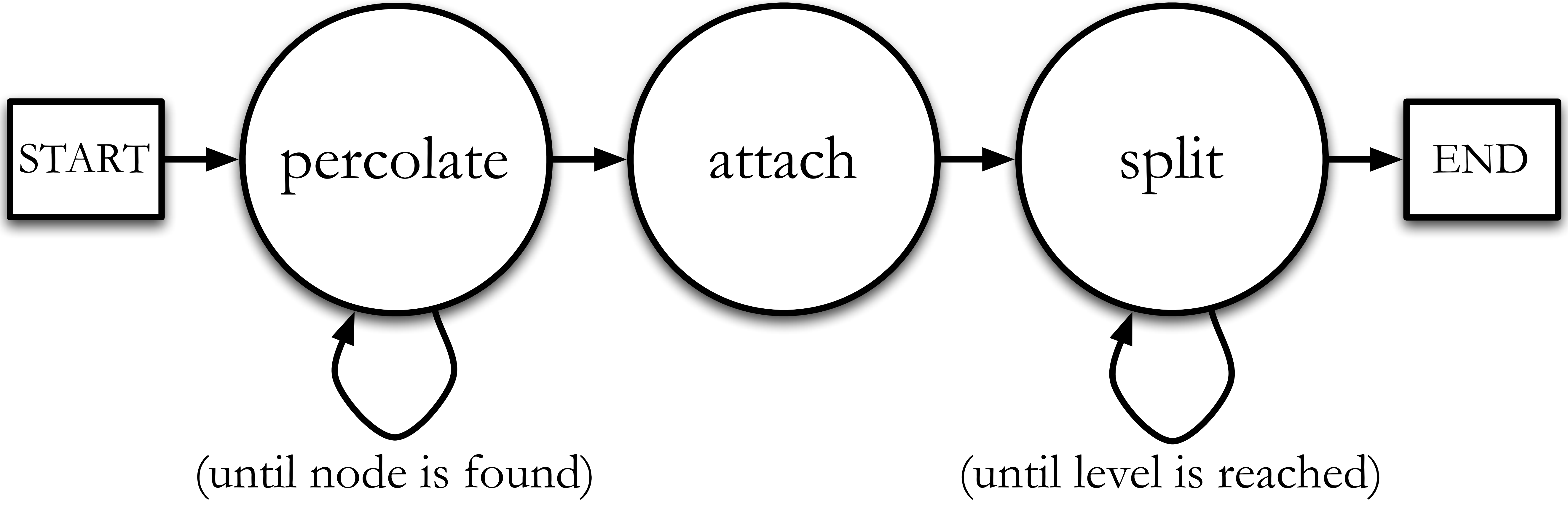
3. **Split**(2)



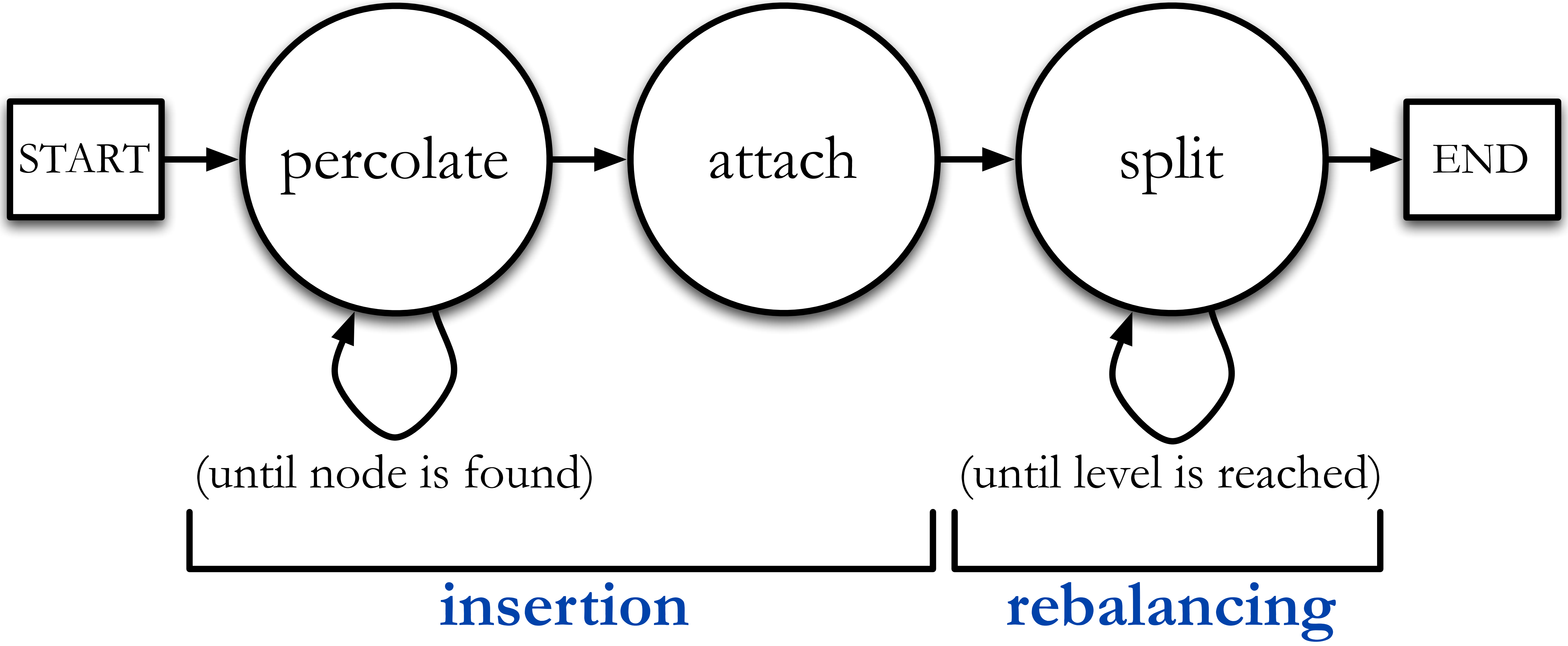
Skip Trees - Insertion



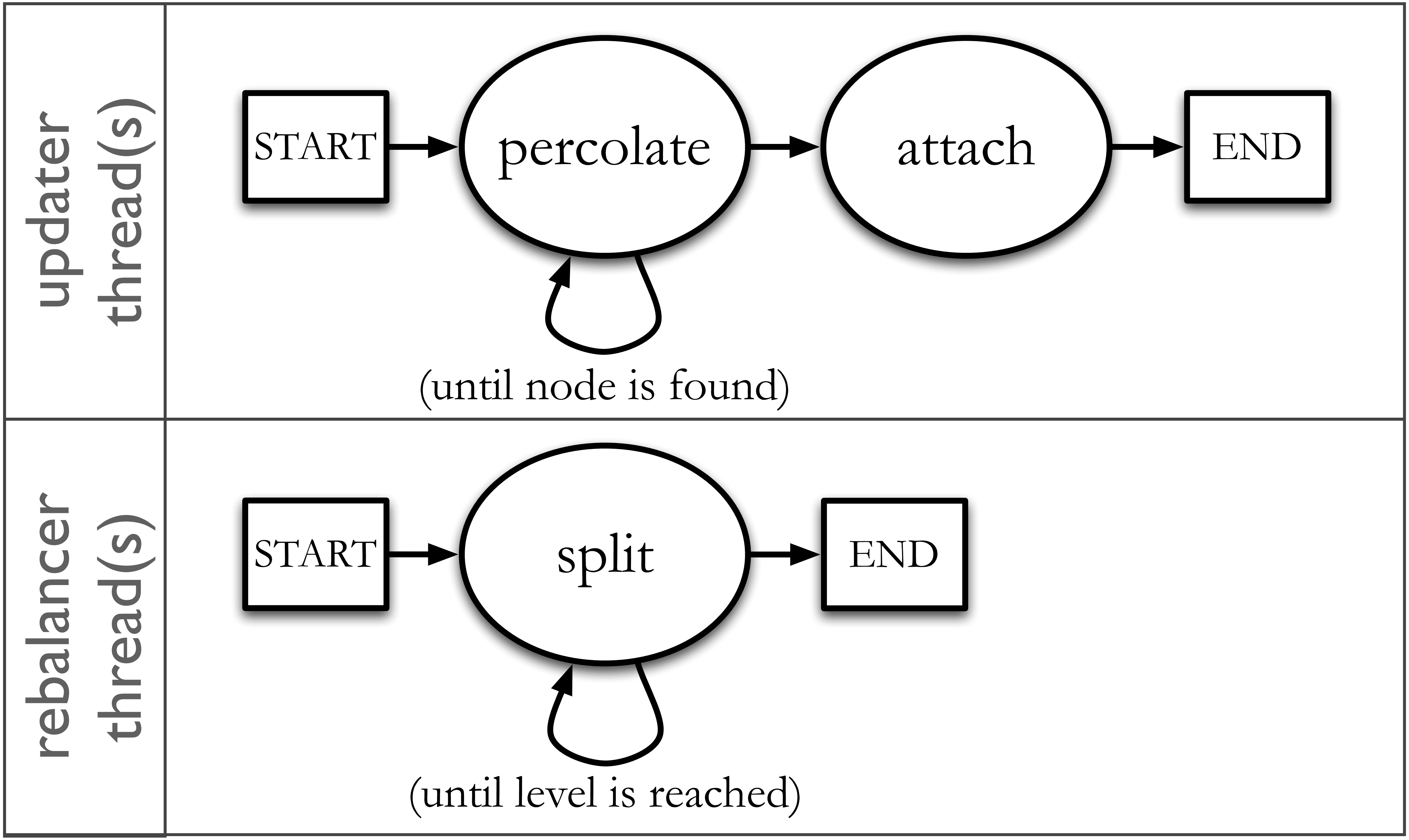
Concurrent Insertion



Concurrent Insertion



Concurrent Insertion



Concurrent Insertion

approach summary

- updater and rebalancer threads run in parallel
- several rebalancer threads can coexist
- percolate and split have local scope
- concurrent deletion is similar to insertion

Project Plan

- study the effects of
  - # of reader threads
  - # of updater threads
  - # of rebalancer threads

References

1. Xavier Messeguer. Skip trees, an alternative data structure to skip lists in a concurrent approach. *RAIRO Theoretical Informatics and Applications*, 31(3):251–269, May 1997.

2. Otto Nurmi and Eljas Soisalon-Soininen. Chromatic binary search trees: a structure for concurrent rebalancing. *Acta Informatica*, 31(6):547–557, September 1996.

3. William Pugh. Skip lists: A probabilistic alternative to balanced trees. *Communications of the ACM*, 33(6):668–676, June 1990.