2019-02-14 Skip Lists

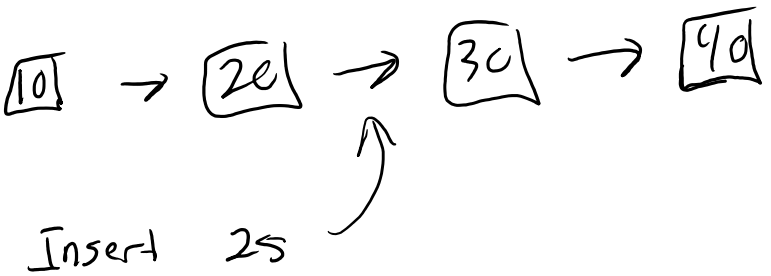
Thursday, February 14, 2019 9:01 AM

- Downside of linked lists is that they don't support random access
  - Finding, adding, and removing elements requires you to start at the front
- In last class, we used a "last accessed" pointer to speed up iteration
  - Made iteration *much* faster
  - Downside: if we try to retrieve something physically before our "last access", we again have to start at the front of the list -> back to being slow
- The observation that linked lists become much faster with additional "cache pointers" is not very new.
  - The question then become: how many cache pointers do we need? And where should they be placed within the linked list?
- Suggestion: place pointer 1/2 way through LL
  - Problem: where is 1/2 way? What happens when 1/2 way moves
    - Backwards is particularly bad because we'd have to reset from the front (SLOW!)
- Solution to the problem of "where to place pointers" is addressed in the Skip List data structure
  - Create "layers" of pointers.
    - Each layer points to data that is spatially farther away
    - Placement of pointers in these layers is handled randomly

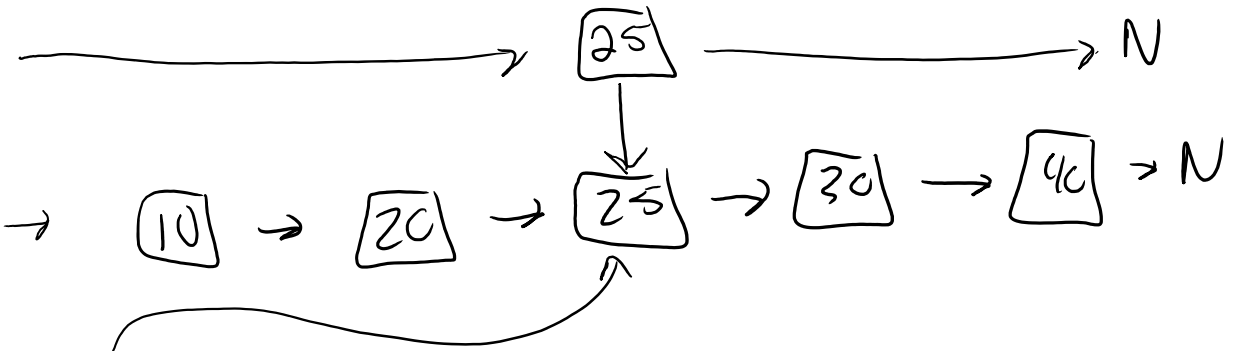
General Idea of Skip Lists

- Unlike normal linked lists, Skip Lists *must* be sorted
- On an insert of new data, flip a coin.
  - If TRUE, turn this element into a pointer at layer N.
  - While coin flip continues to be true, add element to successively higher levels of pointers

Graphical Example



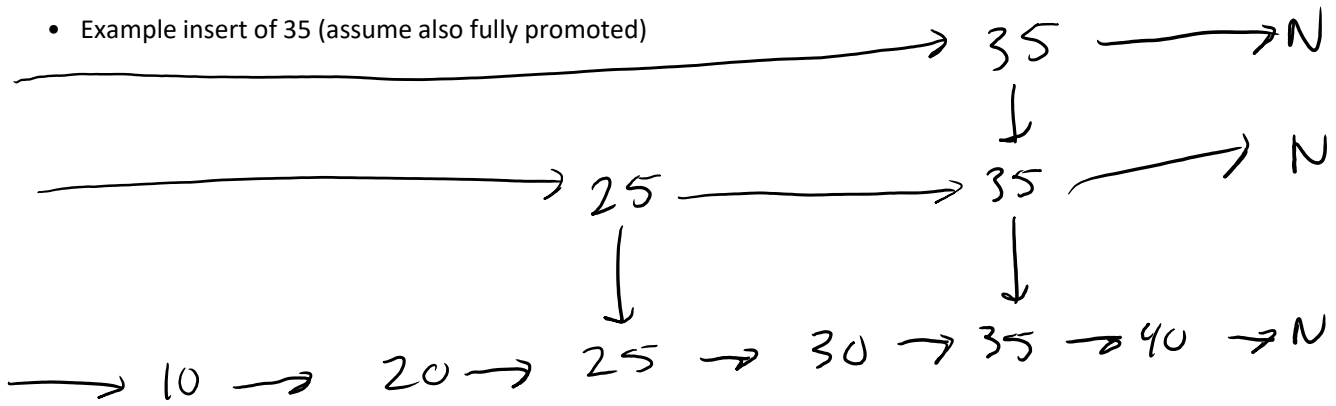
- Step #1: Insert 25 as normal



- WHILE coin flip continues to be TRUE or value is promoted to N+1 highest level
- Say, coin flip is TRUE

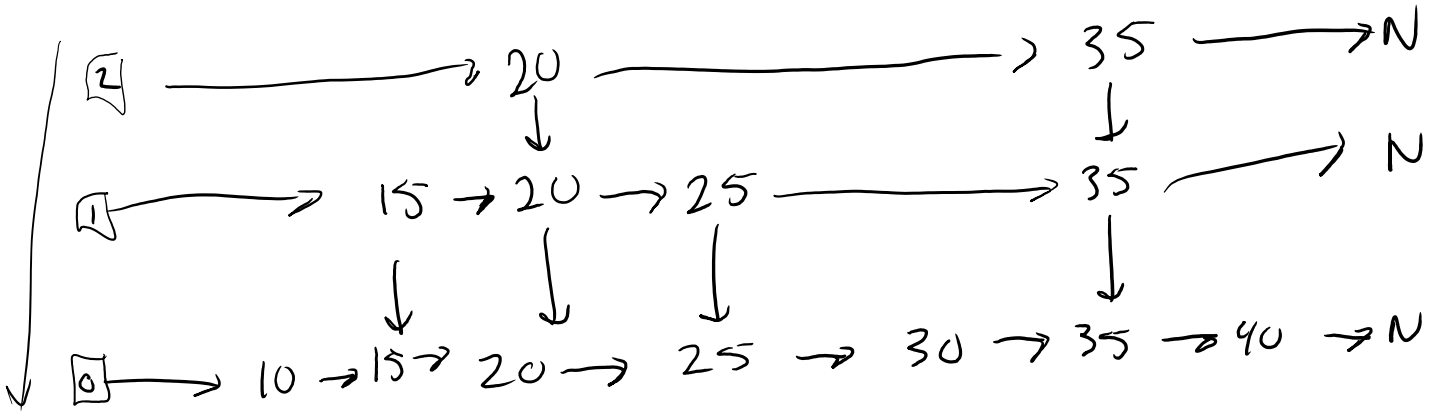
- WHILE coin flip continues to be TRUE or value is promoted to N+1 highest level
- Say, coin flip is TRUE

- Example insert of 35 (assume also fully promoted)

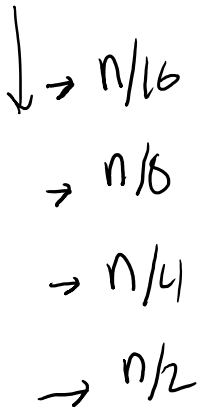


- Performing operations on a Skip List start at the top layer and work down as needed

- Insert 15, with a single promotion



- If our base layer has N elements, on average how many elements does the layer immediately above it have?
  - Each layer will, on average, have 1/2 the number of elements



→ N