CS180 - Recap of our semester

Note Title 12/12/2011

Announcements

- HW is due
- Review session: Friday at 4pm in Linux lab (check webpage Thursday in case location changes)
- Final Monday at noon
- Office hours: Wed. morning, Friday morning

Data Structures Covered

- stacks
- queue
- vectors
- lists
- heaps
- tree structures
  - BST
    - AVL trees
  - Huffman trees
  - treaps
  - hashing
    - graphs

searching
sorting

Data Structures

Some data structures have limited functionality, but as a result are extremely efficient.

Ex. - stack & queue

- graph representations (space vs. speed)

"Full-featured" data structures

More versatile data structures have trade-offs:
to get something faster, you sacrifice speed in another area.

Ex:

vectors		lists
O(1) - operator [ ]	→	O(n)
O(n) - insert	→	O(1)

AVL trees: O(log n)

Randomized or Expected

Some work well in practice, but have no theoretical guarantees.

↑ worst case

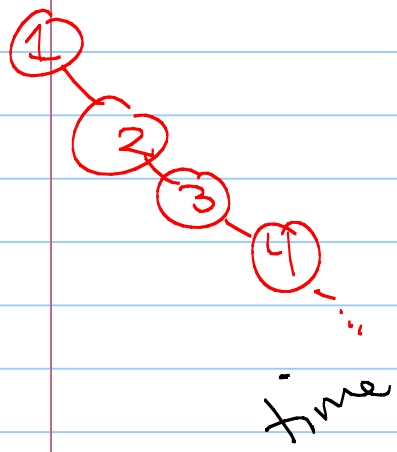
Ex:

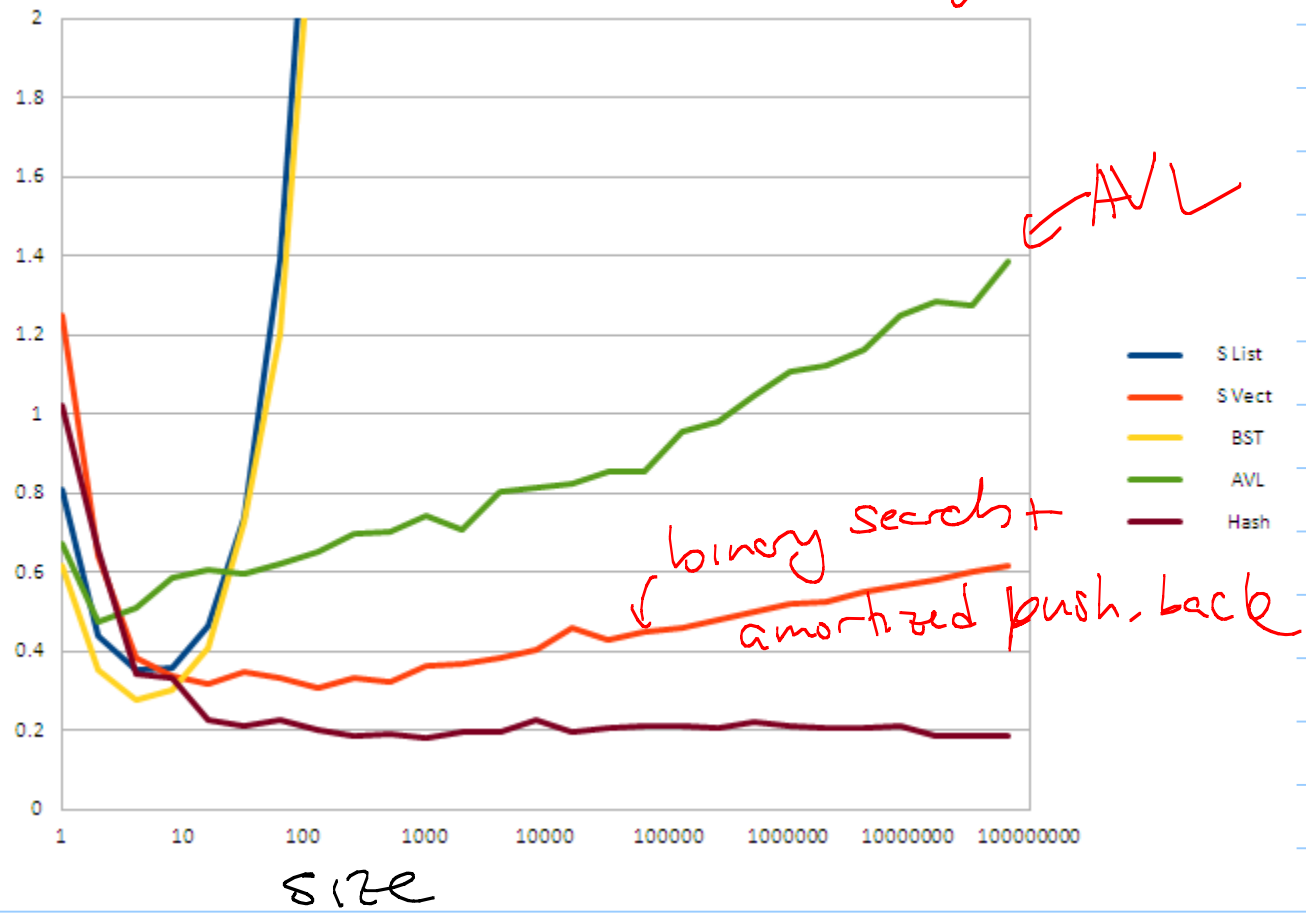
- hashing
- treap – random priority
- quicksort

So which is best?

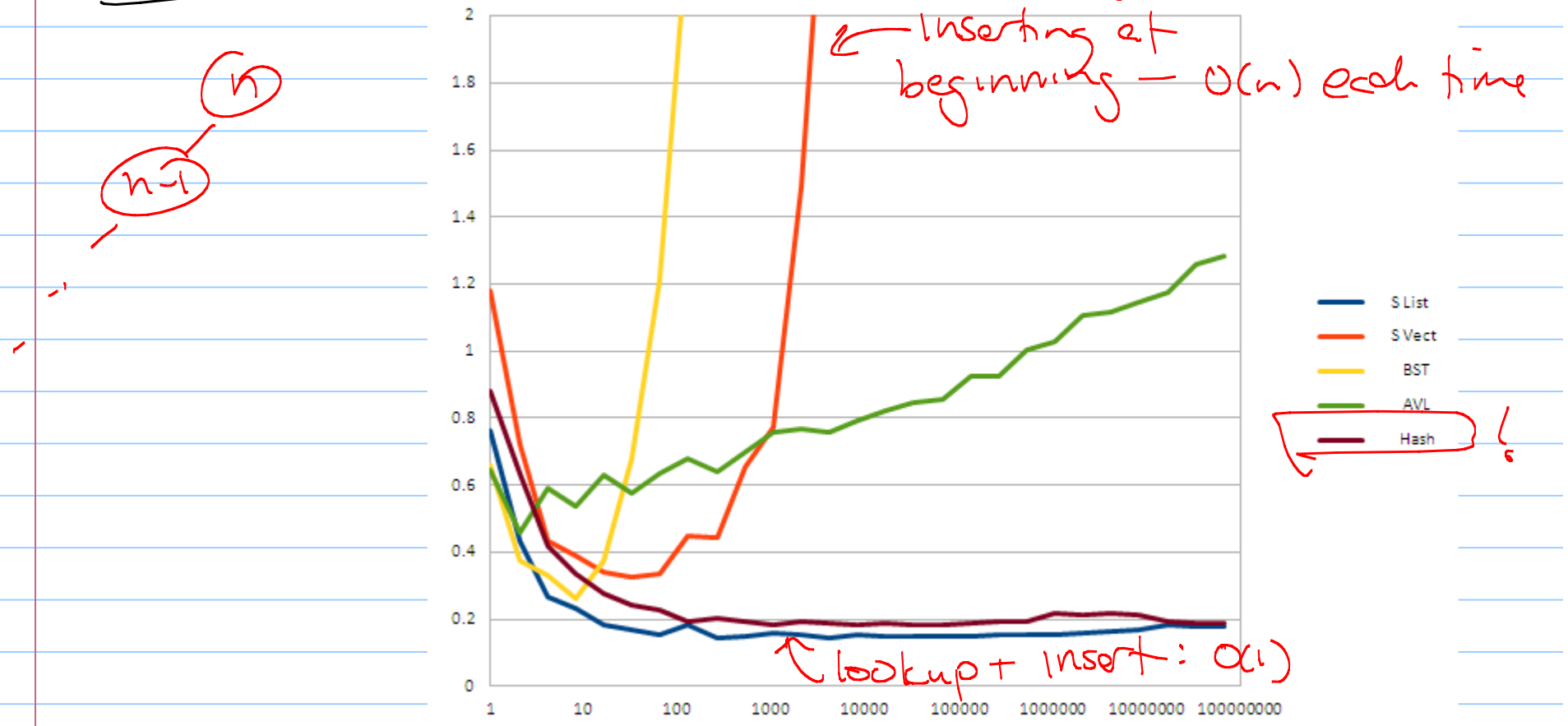
Ans: Depends!

In-order insertions: 1, 2, 3, ..., n.



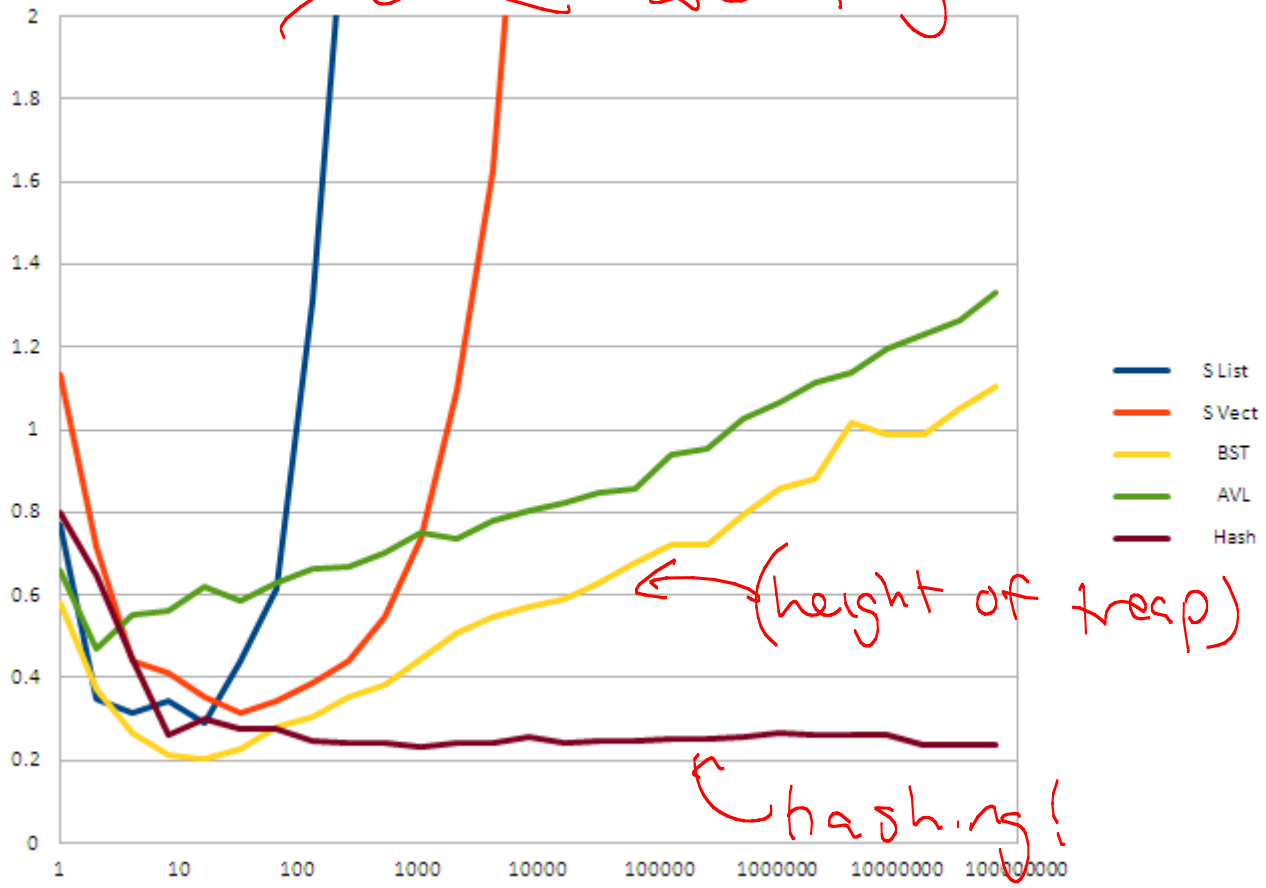


Reverse Order inserts : n, n-1, n-2, ... 1



Random Inserts

both list & vectors aren't good



(height of treap)

Thashing!

Note: hashing does lack extra functionality, so not always the right choice

Questions:

insert(Q, 2)

treap:

unique!