CS6033 Lecture 4 Slides/Notes

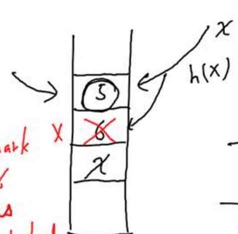
Hash Tables & Search Trees (Notes, Ch 11 (skip Secs. 11.3.4, 11.3.5 and 11.5), Handouts for Search Trees)

By Prof. Yi-Jen Chiang
CSE Dept., Tandon School of Engineering
New York University

1

Open Addressing: i: probing #. k: key

$\alpha \leq 1$. $h(k, i)$ $h(\langle k, i\rangle)$ $i = 0, 1, 2, \cdots$



mark 6 as deleted but still keep 6.

M1: Linear probing. $h(\langle k,i\rangle) = (h'(k) + i) \bmod m$

M2: Quadratic " $h(\langle k,i\rangle) = (h'(k) + c_1 i + c_2 i^2) \bmod m$ for some consts. c_1, c_2.

M3: Double hashing. $h(\langle k,i\rangle) = (h_1(k) + i\, h_2(k)) \bmod m$ for some hash functions $h_1()$, $h_2()$

(M2 was in 3rd Ed. but omitted in 4th Ed.)

Use the assumption of independent uniform permutation hashing (also called uniform hashing)

Assume: Each of the $m!$ permutations (on $0, 1, \cdots, (m-1)$) are equally likely to occur during the probing.

2

Thm: In open addressing the expected time for an unsuccessful search is $\leq \frac{1}{1-\alpha}$ $\left(\Theta(\alpha+1)\right)$

Pf: Let X be a random var. for the # of probes in an unsuccessful search.

A_j: event that the j-th probe is to an occupied slot

$\Pr\{X \geq i\} = ?$

$\Pr\{A_1\} = \frac{n}{m}$ $\quad \Pr\{A_2 | A_1\} = \frac{n-1}{m-1}$ $\left(= \frac{\Pr\{A_2 \cap A_1\}}{\Pr\{A_1\}}\right)$

$\Pr\{A_2 \cap A_1\} = \Pr\{A_1\} \cdot \Pr\{A_2 | A_1\} = \frac{n}{m} \cdot \frac{n-1}{m-1}$, etc.

Event $X \geq i$
$= A_1 \cap A_2 \cap \cdots \cap A_{i-1}$
= Event that the first $(i-1)$ probes all go to occupied places.

(Remark: The last probe goes to an empty place)

$$\boxed{\Pr\{X \geq i\}} = \underbrace{\underset{\text{1st}}{\left(\frac{n}{m}\right)} \cdot \underset{\text{2nd}}{\left(\frac{n-1}{m-1}\right)} \cdots \frac{n-(i-2)}{m-(i-2)}}_{(i-1) \text{ terms}}$$

$$\boxed{\leq \alpha^{i-1}}$$

e.g. $\frac{n}{m} = \frac{3}{4} = 0.75$ big
$\frac{n-1}{m-1} = \frac{2}{3} = 0.66$
$\vdots$
small

$$E[X] = \sum_{i=1}^{\infty} i \cdot \Pr\{X = i\} = \sum_{i=1}^{\infty} i \cdot \left(\Pr\{X \geq i\} - \Pr\{X \geq (i+1)\}\right)$$

$\{i, i+1, i+2, \cdots\}$ $\quad \{i+1, i+2, \cdots\}$

i remains

$$= \sum_{i=1}^{\infty} i \Pr\{X \geq i\} - \underbrace{\sum_{i=1}^{\infty} i \cdot \Pr\{X \geq i+1\}}_{= \sum_{i=1}^{\infty} (i-1) \cdot \Pr\{X \geq i\}}$$

$$= \sum_{i=1}^{\infty} \underbrace{\left(i-(i-1)\right)}_{=1} \cdot \Pr\{X \geq i\} = \sum_{i=1}^{\infty} \Pr\{X \geq i\} \leq \sum_{i=1}^{\infty} \alpha^{i-1} = 1 + \alpha + \alpha^2 + \cdots$$

$$= \frac{1}{1-\alpha} \quad \#$$

Collorary: Expected time for insert is $\leq \frac{1}{1-\alpha}$

Pf: Same time as unsuccessful search.

Thm: Expected time for a successful search in open addressing is $\le \frac{1}{\alpha} \ln \frac{1}{1-\alpha}$.

Pf: From the corollary, the insert time for the $(i+1)$st item is $\frac{1}{1-\alpha_i}$

where $\alpha_i = \frac{i}{m}$ is the load factor right before this insertion.
(since there are i items in the hash table)

This insert time is the same as the search time for this $(i+1)$st item.

$\Rightarrow$ search time for the $(i+1)$st item is $\frac{1}{1-\alpha_i}$.

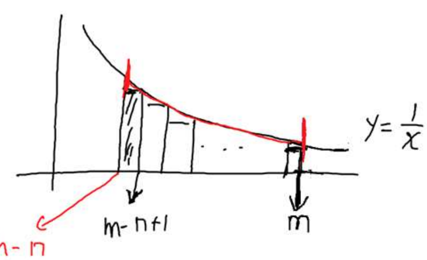
Let X be a random variable for # of probes in a successful search

$$E[X] \le \frac{1}{n} \sum_{i=0}^{n-1} \frac{1}{1-\alpha_i} = \frac{1}{n} \sum_{i=0}^{n-1} \frac{1}{1-\frac{i}{m}} = \frac{1}{n} \sum_{i=0}^{n-1} \frac{m}{m-i} \quad \left(\begin{array}{cc} i+1: 1 \to n & i: 0 \to n-1 \end{array} \right)$$

$$= \frac{m}{n} \sum_{i=0}^{n-1} \frac{1}{m-i} = \frac{1}{\alpha}\left(\frac{1}{m} + \frac{1}{m-1} + \cdots + \frac{1}{m-n+1}\right) = \frac{1}{\alpha} \sum_{k=m-n+1}^{m} \frac{1}{k}$$

5

$$E[X] \le \frac{1}{\alpha} \sum_{k=m-n+1}^{m} \frac{1}{k}$$



$$\le \frac{1}{\alpha} \int_{x=m-n}^{m} \frac{1}{x}\,dx$$

$$= \frac{1}{\alpha} \ln x \Big|_{x=m-n}^{m} = \frac{1}{\alpha}\left(\ln m - \ln(m-n)\right) = \frac{1}{\alpha} \ln \frac{m}{m-n}$$

$$= \frac{1}{\alpha} \ln \frac{1}{1-\frac{n}{m}} = \frac{1}{\alpha} \ln \frac{1}{1-\alpha}. \quad \#$$

6

New Topic: Search Trees

- Discussed the binary search tree handout ``1-BinarySearchTrees.pdf'', in particular, deletions on slides 8 and 9.

7

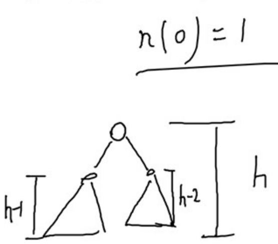
Search Trees: AVL-Trees

- Discussed the AVL-tree handout ``2-AVLTrees.pdf'' --- definition, tree height, rebalancing via single & double rotations. (See also the slides next.)

8

Def: $n(h)$ = min # nodes in an AVL-tree with height h.

$\underline{n(0) = 1}$ $\qquad$ $\underline{n(1) = 2}$



$n(h-1) \geq n(h-2)$

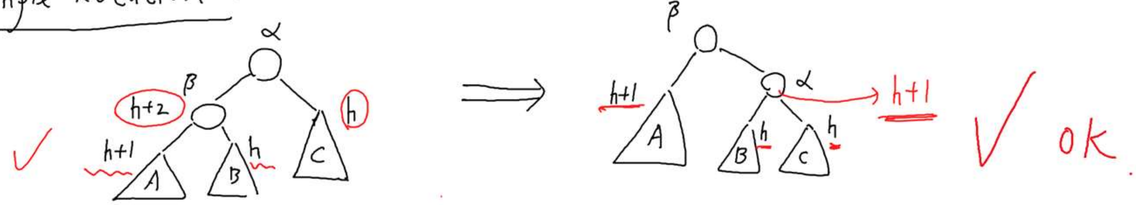
$$\begin{aligned} n(h) &= n(h-1) + n(h-2) + 1 \\ &\geq 2 \cdot n(h-2) \\ &\geq 2\left[2\, n(h-4)\right] = 2^2 \cdot n(h-2\cdot 2) \\ &\geq 2^2\left(2\, n(h-4-2)\right) = 2^3 \cdot n(h-2\cdot 3) \\ &\geq \cdots \\ &\geq 2^i\, n(h-2\cdot i) \geq \cdots \geq 2^{h/2}\, n\left(h-2\cdot\tfrac{h}{2}\right) \quad (i \leftarrow h/2) \\ &= 2^{h/2} \cdot c \end{aligned}$$

$\Rightarrow n \geq n(h) \geq 2^{h/2} \cdot c \Longrightarrow h/2 \leq \log_2(n/c)$. $h = O(\log n)$

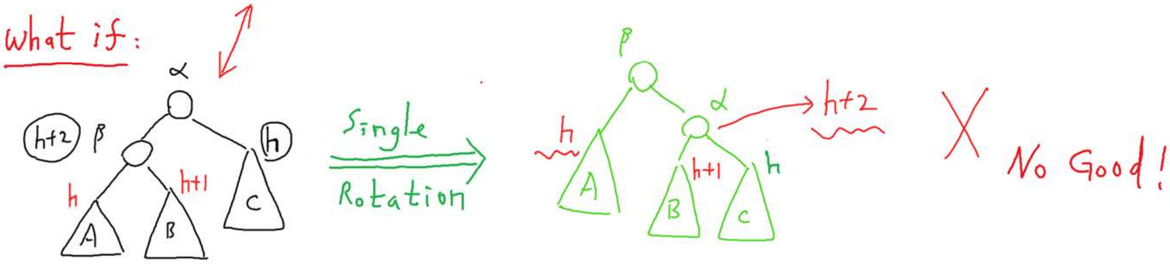
This is to prove that an AVL-tree with n nodes has height h = O(log n).

9

Single Rotation:

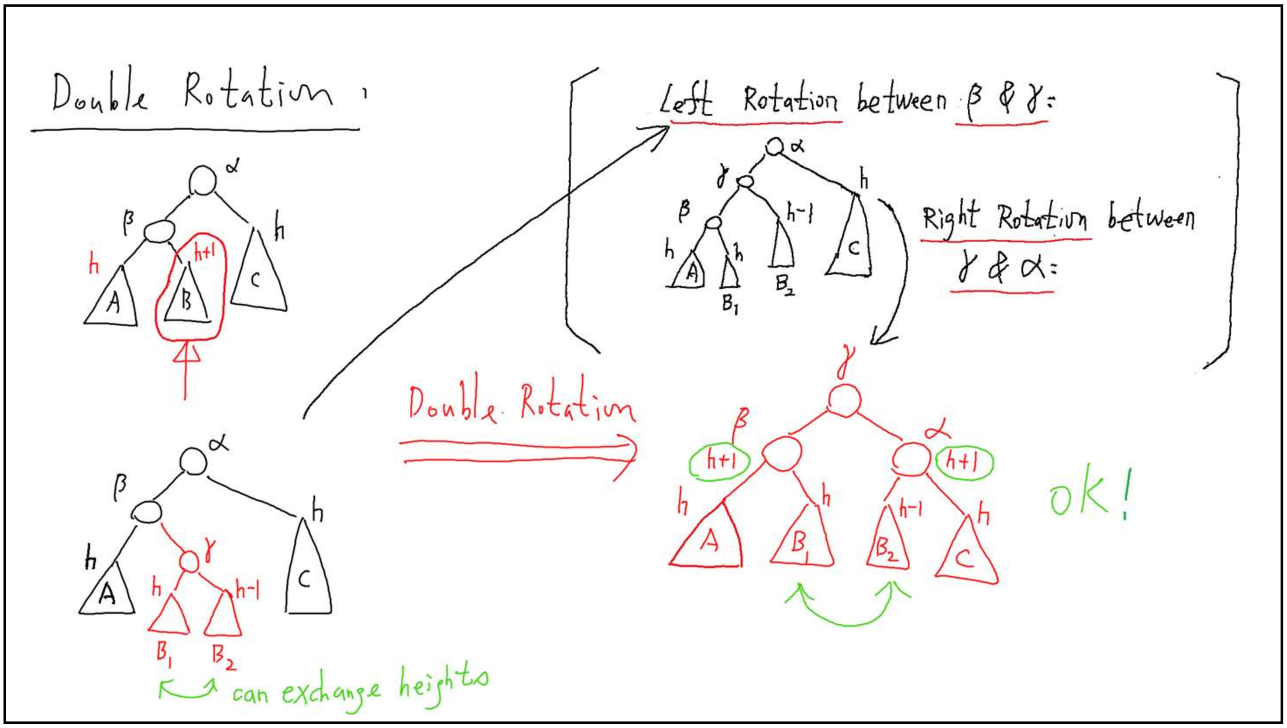


what if:



We need double rotation instead!

10



11

AVL-Trees & (2,4)-Trees

- AVL-Trees: There are symmetric cases for single rotation and double rotation. See slides 7 and 8 of the handout ``2-AVLTrees.pdf.''
- (2,4)-Trees: Discussed the handout ``3-24Trees.pdf'': inorder traversal, (2,4)-tree definition, tree height, 2-pass insertion (issue of overflow & op of split), 2-pass deletion (issue of underflow & ops of transfer, fusion). See slides 3 – 13.

12