Data Structures and Algorithms

https://eduassistpro.github.io/

Lecture 10_2: Bina edu_assist_pro

(Average Case Analysis)

Overview

- Binary Search Trees
 - Searching • Best Case / Worst Case Analysis

 - Average https://eduassistpro.github.io/

Runtime Analysis

Analyzing the runtime, we may have different perspectives:

- Worst cases an alysist (per ecs refarm default case)
- Best case an
- Average cas

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Notation:

i: problem instance

T(i): runtime on i

 I_n : set of instances of size n.

Worst Case Analysis

$$T_w(n) = \max\{T(i) : i \in I_n\}$$

Assignment Project Exam Help

Example:

https://eduassistpro.github.io/ Find for binary search trees: $T_w(n) = \Theta(n)$ Add WeChat edu_assist_pro

- Tree may degenerate to a list
- Tree has height n-1
- We want to find the element of height n-1

Best Case Analysis

$$T_b(n) = \min\{T(i) : i \in I_n\}$$

Assignment Project Exam Help

Example: https://eduassistpro.github.jo/ Find for binary search trees: $T_b(n) = O(1)$ Add WeChat edu_assist_pro

The element is at the root of the tree

Analysis for height of a binary search tree:

• Worst case: $\Theta(n)$ • Best case: $\Theta(\log n)$ • https://eduassistpro.github.io/

Average Case Analysis

$$T_a(n) = \frac{1}{|I_n|} \sum_{i \in I_n} T(i)$$

Average Ansignment Perojecto Examinatelp

https://eduassistpro.github.io/

Average time for find (degenerated tree)

Assume: T is generated to a list and consists of elements 1, 2, ..., n.

elements 1, 2, ..., n. Assignment Project Exam Help Input for find: $i \in \{1, \ldots, n\}$

https://eduassistpro.github.io/

Average time to find an el Tchosen uniformly at random is

$$\frac{1}{n} \cdot (1+2+\ldots+n) = (n+1)/2$$

Average time for find (balanced tree)

Assume:

- T is perfectly balanced. Assignment Project Exam Help
- $n = 2^k 1$ ele

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Observation:

- There are 2^i elements at depth i, $0 \le i \le k-1$.
- Time to find element at depth i is i+1.

Time to find an element in T chosen uniformly at random is

Assignment Project Exam Help

$$\frac{1}{n} \cdot \sum_{i=0}^{k-1} (i + h_{t}) s:/2 i$$
duassistpro.github.io/

$$= \frac{1}{n} \cdot (k \cdot 2^{k-1} + (k-1) \cdot 2^{k-2} + \dots + 1 \cdot 2^{0})$$

$$=\frac{1}{n}\cdot(\qquad 2^{k-1}+2^{k-2}+\ldots+2^0$$

$$\begin{array}{c} \text{Assignment Projects Exam Help}\\ +2^k+2^k+\ldots+2^1\\ +\text{https://eduassistpro.github.io/}\\ +2^k+2^k+\ldots+2^2\\ \text{Add WeChat edu_assist_pro}\\ & \cdots\\ +2^{k-1}+2^{k-2}\\ +2^{k-1}) \end{array}$$

Use geometric series:

$$\sum_{i=0}^{k} 2^{i} = 1 + 2 + 4 + \dots 2^{k} = 2^{k+1} - 1$$

We get:

Assignment Project Exam Help

$$= \frac{1}{n} \cdot ((2^k - 2^0) + \frac{k^k}{n^k} \operatorname{ps:})^{\frac{1}{2}} \operatorname{edu}(3^k \operatorname{sistp})^{\frac{2}{2}} \operatorname{glthub}(3^k - 2^{k-1}))$$

$$= \frac{1}{n} \cdot (k \cdot 2^k - \operatorname{Add}(3^k \operatorname{wech}) \operatorname{edu_assist_pro})$$

$$= \frac{1}{n} \cdot (\log(n+1) \cdot (n+1) - n)$$

$$= (1 + \frac{1}{n}) \log(n+1) - 1$$

Theorem

Theorem: The average time to find an element in a perfectly balanced binary tree with $n = 2^k-1$ elements is $(1 + \frac{1}{n}) \log(n + 1) - 1$ https://eduassistpro.github.io/

Average Case for random insertion

 Assume that the items to be inserted are in random order.

Assignment Project Exam Help

 We may be as small depth (does not d https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Question:

 What is the average time to find an element in such a tree?

Permutations of n elements

Assume that we have a set of n elements

Consider all permutations of these elements Assignment Project Exam Help

There are n! p

https://eduassistpro.github.io/

Add WeChat edu_assist_pro Example: Set {1, 2, 3}

Permutations:

(1,2,3), (1,3,2), (2,1,3), (2,3,1), (3,1,2), (3,2,1)

Analysis

In our analysis:

- we average over the different permutations for building the ignment earlier Help
- all queries fo https://eduassistpro.github.io/

Add WeChat edu_assist_pro
Formally, we consider "dou ted value" with
respect to:

- the order of elements inserted
- the element we query

Cost of a search tree

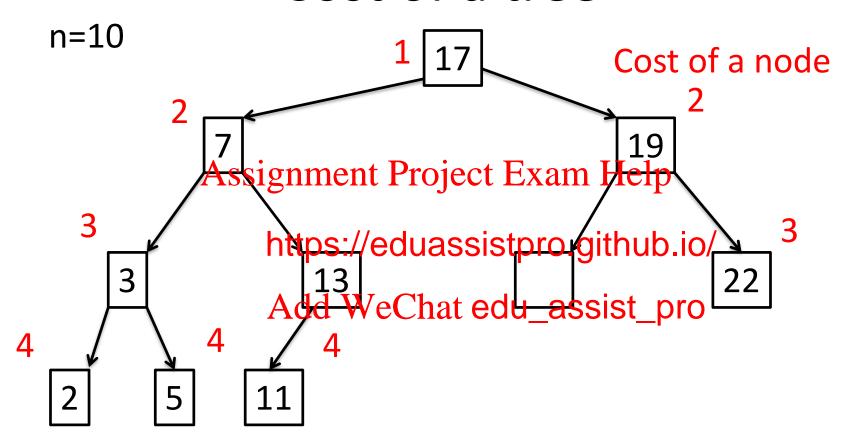
c(v): number of nodes on the path from the root to v.

Assignment Project Exam Help Cost of a tree

https://eduassistpro.github.io/
$$C(T) = \sum_{v \in T} c(v)$$
 Add WeChat edu_assist_pro

Average search cost of a tree T: C(T)/n

Cost of a tree



Cost of the tree C(T) = 1+2+2+3+3+3+3+4+4+4=29

Average search time for T: C(T) / n = 29 / 10 = 2.9

Average costs of a tree

Let E(n) be the average cost of tree with n elements.

Assignment Project Exam Help

Recursion:

https://eduassistpro.github.io/

$$E(0) = 0$$

$$E(1) = 1$$

$$E(n) = n + \frac{1}{n} \sum_{i=1}^{n} (E(i-1) + E(n-i))$$

Recursive Formula

i-1 elements go into n-i elements go into the left subtree the right subtree Assignment Project Exam Help $E_{i\neq e}^{n}$ $(E(i-1) + E(\gamma))$ Add WeChat edu_assist_pro ith equal Root lies on every probability the root path to a node

Solve Recursion

- Recursive Formula seems to be complicated.
- Is it worth the effort?
 Assignment Project Exam Help

Reasons for d https://eduassistpro.github.io/

- Result is interesting Chat edu_assist_pro
- Math tricks can often be used
- Similar analysis gives average case results for the Quicksort algorithm.

Solving Recursion

$$E(n) = n + \frac{1}{n} \sum_{i=1}^{n} (E(i-1) + E(n-i))$$
 contains E(0), E(1), ..., E(n-1), and Help

https://eduassistpro.github.io/

First step:

 Add WeChat edu_assist_pro
 Get a recursive formula hat o hat only depends on E(n-1).

Consider
$$n \cdot E(n) - (n-1)E(n-1)$$

Assignment Project Exam Help

This implies th the same https://eduassistpro.github.io/

$$n \cdot E(n) = n^2 + \sum_{i=1}^{\text{Add}} (E(i-1) + E(n-i))$$

$$= n^2 + 2 \cdot (E(1) + E(2) + \dots + E(n-1))$$

$$(n-1) \cdot E(n-1) = (n-1)^2 + \sum_{i=2}^n (E(i-1) + E(n-i))$$

$$= (n-1)^2 + \sum_{i=2}^n (E(1) + E(2)) + E(2) + E(2)$$

$$= (n-1)^2 + \sum_{i=2}^n (E(i-1) + E(n-i))$$

$$n \cdot E(n) - (n_{\mbox{Add}} \cdot E(n_{\mbox{Add}} \cdot E$$

$$n \cdot E(n) - (n+1) \cdot E(n-1) = 2n-1$$

Assignment Project Exam Help Divide by n(n+

$$\frac{1}{n+1} \cdot E(n) \frac{1}{\text{Add}^n \text{WeChat edu_assist}} \underbrace{\frac{2n-1}{2n-1}}_{\text{Add}^n \text{WeChat edu_assist}} \underbrace{\frac{2n-1}{2n-1}}_{\text{Neck edu_assist}}$$

Consider:

$$Z(n) = \frac{1}{n+1} \cdot E(n)$$

$$\begin{split} Z(n) &= Z(n-1) + \frac{2n-1}{n(n+1)} \\ &= \underset{\text{assignment Project Exam Help}}{\text{Assignment Project Exam Help}} \\ &= Z(n-2) + \frac{2(n-1)-1}{n(n+1)} + \frac{2n-1}{n(n+1)} \\ &= Z(n-1) + \frac{2(n-1)-1}{n(n+1)} \\ &= Z(n-1) +$$

Use:
$$\frac{1}{i(i+1)} = \frac{1}{i} - \frac{1}{i+1}$$

Then we get: Assignment Project Exam Help

$$Z(n) = \frac{2}{\text{https2//eduassistp2o.github.io/}} \frac{1}{i+1}$$
 Add WieChat edu_assist_pro

$$-\sum_{i=1}^{n} \frac{1}{i} + \sum_{i=1}^{n} \frac{1}{i+1}$$

$$= 2n - 2n + 2\sum_{\text{Assignment Project Exam Help}}^{n} \frac{1}{i} - 1 + \frac{1}{n}$$

$$= 2\sum_{i=1}^{n} \frac{1}{i} - \frac{1}{2} \frac{1}{i} - \frac{1}{2} \frac{1}{i} - \frac{1}{2} \frac{1}{i} \frac{1}{i} \frac{1}{i} - \frac{1}{2} \frac{1}{i} \frac{1}{i} \frac{1}{i} - \frac{1}{2} \frac{1}{i} \frac{1}$$

$$= 2 \cdot H(n) - 3 + \frac{3}{n+1}$$

Harmonic sum
$$H(n) = \sum_{i=1}^{n} \frac{1}{i}$$

Data Structures and Algorithms

Remember:
$$Z(n) = \frac{1}{n+1} \cdot E(n)$$

$$E(n) = \text{Assignment Z (oject Exam Help})$$

$$= 2(n+1) \cdot H(n) - 3(n+1) + 3$$
Add WeChat edu_assist_pro

Average Cost for Find

Average cost for find after random insertion:

$$E(n)/n = A \text{Signment Project}(\text{Exam Belp} \frac{n+1}{n} + \frac{3}{n})$$
Using: $\ln(n+1) \leq H(n) \leq \ln n + 1$
Add WeChat edu_assist_pro
we get
$$E(n)/n = 2 \cdot \ln n - O(1) = (2 \ln 2) \cdot \log n - O(1)$$

$$\approx 1.386 \cdot \log n$$

Theorem

Theorem: The insertion of n randomly chosen elements leads to a Binary Search Tree whose expected time Assignment Project Exam Help d'operation is

 $(2 \ln 2) \cdot lhttps://edua/stistpro.lgiff@6.io/<math>log n$ Add WeChat edu_assist_pro

Runtimes for Binary Search Tree

Find, insert, remove:

Worst case: $\Theta(n)$ Assignment Project Exam Help Best case: $\Theta(\log n)$ https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Aim: Time O(log n) in the worst case