CSE515 Advanced Algorithms
Lecture 27: Tail Inequalities

Antoine Vigneron
antoine@unist.ac.kr

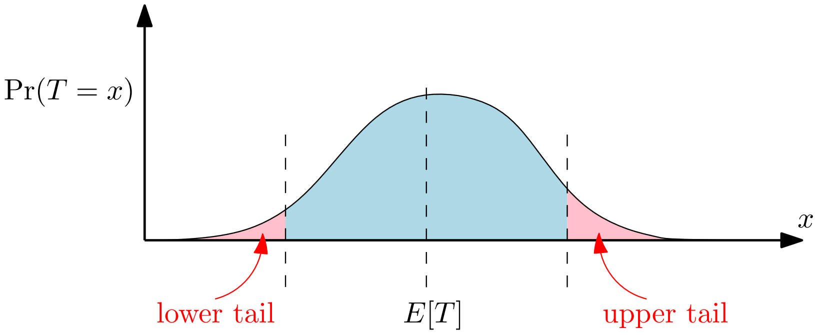
Ulsan National Institute of Science and Technology

July 27, 2021

Introduction

- In previous lectures, we studied the *average* running time of some algorithms.
- In this lecture, we will show how to estimate how often the running time deviates substantially from this average.
- References:
  - Section 13.9 of Algorithm Design by Kleinberg and Tardos.
  - Rao Kosaraju's lecture notes on tail inequalities.

Tail Inequalities



- So far we only studied the average running time $E[T]$ of some randomized algorithms.
- We now want to give better guarantees: we want to show that T very seldom is much larger than $E[T]$. In other words, we want to show that the *upper tail* is small.

First Approach

Example

Suppose the average running time of your randomized algorithm is 1h. What can you say about the probability that it runs for more than 10h?

- Answer in lecture notes.

Markov's Inequality

Theorem (Markov's inequality)

If X is a nonnegative random variable, $\mu = E(X)$ and $c > 0$, then

$$\Pr(X \geqslant c\mu) \leqslant \frac{1}{c}.$$

- Proof done in class. Ref: R. Kosaraju's lecture notes.
- It is often formulated $\Pr(X \geqslant a) \leqslant E(X)/a$, but for us the formulation above will be more convenient.

Example

QUICKSORT runs in expected time at most $Cn \log n$ for some constant C. Then the probability that it takes more than $100Cn \log n$ time is $\leqslant 1\%$.

Chernoff Bounds

- We now present *Chernoff bounds*, which often provide better estimates than Markov's inequality, but require stronger conditions on the random variables.

Theorem (Chernoff bound for upper tail)

Let $X_1, \ldots, X_n$ be independent random variables taking values in $\{0, 1\}$. Let $X = X_1 + \cdots + X_n$ and $\mu = E(X)$. Then for any $\delta > 0$

$$\Pr[X \geqslant \mu(1+\delta)] \leqslant \left(\frac{e^\delta}{(1+\delta)^{1+\delta}} \right)^\mu .$$

- Proof done in class. Ref: textbook and Kosaraju's lecture notes.

Chernoff Bounds

- The bound in previous slide allows to estimate how often X is much larger than its average μ.
- The bound below is for the case where X is much smaller than μ.

Theorem (Chernoff bound for lower tail)

Let X be defined as above. Then, for any $0 \leqslant \delta \leqslant 1$

$$\Pr[X \leqslant (1-\delta)\mu] \leqslant e^{-\frac{\mu\delta^2}{2}}.$$

- Proof is not covered. Similar as previous theorem.

Chernoff Bounds

Example

You toss 200 coins. What do you know about the probability of getting no more than 50 Heads?

- $\Pr[X \leqslant 50] \leqslant e^{-100/8} \approx 3.7 \times 10^{-6}$.

Example

You generate $2n$ independent random bits. How likely is it that they sum to no more than $n - 10\sqrt{n}$?

- $e^{-50} \approx 1.9 \times 10^{-22}$.

Random Binary Search Tree

Theorem

Let T be a random BST over n nodes $a_1, \ldots, a_n$. Then for each i,

$$\Pr[\text{depth}(a_i) \geqslant 32 \ln n] \leqslant \frac{1}{n^4}$$

- Proof in lecture notes.
- It shows that a node has logarithmic depth with probability at least $\geqslant 1 - 1/n^4$.
- It also shows that insertion and searching in a random BST take $O(\log n)$ time with probability at least $1 - 1/n^4$.
- Terminology: When we have a bound of this type, i.e. a probability $p \geqslant 1 - 1/n^2$ or $p \geqslant 1 - 1/n^5$, we say that it occurs with *high probability*.
- For instance, we can say that insertion into a random BST takes logarithmic time with high probability.

Random Binary Search Tree

- What about the *height* of the tree?
- $h(T) \geqslant 32 \ln n$ iff $\text{depth}(a_i) \geqslant 32 \ln n$ for some i. Therefore:

$$\Pr[h(T) \geqslant 32 \ln n] \leqslant \sum_{i=1}^{n} p(a_i \geqslant 32 \ln n) \leqslant \frac{n}{n^4} = \frac{1}{n^3}$$

Corollary

The height of a random BST over n nodes is at most $32 \ln n$ with probability at least $1 - 1/n^3$.

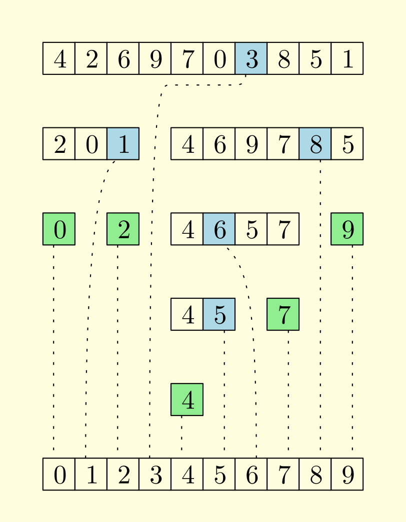
QUICKSORT

Theorem

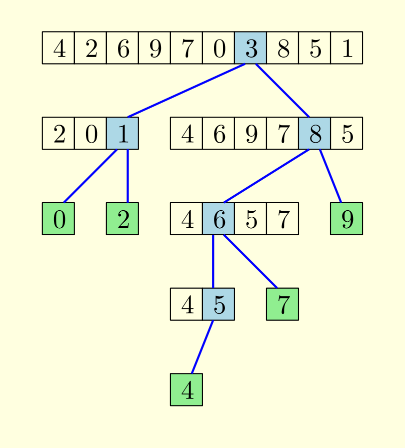
With probability at least $1 - 1/n^3$*, the height of the recursion tree of* QUICKSORT *is at most* $32 \ln n$*. Hence,* QUICKSORT *runs in time* $O(n \log n)$ *with probability at least* $1 - 1/n^3$*. In other words, it runs in time* $O(n \log n)$ *with high probability.*

- We could reprove this directly from Chernoff bounds.
- Instead, we will argue that QUICKSORT and random BST construction are essentially the same, and so we get the same bound.

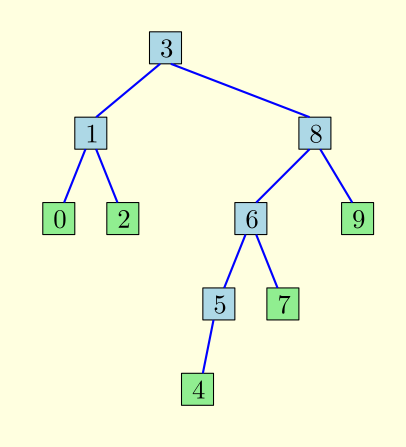
Quicksort



Quicksort



Quicksort



random BST

Quicksort

- So the recursion tree of QUICKSORT, where each node is labeled by its pivot, is a random BST.
- It follows that the recursion depth of QUICKSORT is at most $O(\log n)$ with high probability.
- As QUICKSORT spends time $O(n)$ at each level of its recursion tree, its running time is $O(n \log n)$ with high probability.

Concluding Remarks

- We showed that the randomized algorithms from the previous lectures not only are efficient on average, but very seldom deviate substantially from their average case.
- For instance, in practice, randomized QUICKSORT never exhibits its worst-case behavior, i.e. worst-case quadratic time.
- It is typically the case with randomized algorithms, this is why we usually only analyze their average running time.