CMPSCI 611 Advanced Algorithms **Fall 2000**

Lecture 15: November 2

Lecturer: Micah Adler *Scribe: Raphen Becker and Patrick Deegan*

15.1 Lecture Overview

15.2 Review of Primality Testing

15.2.1 Miller-Rabin

$w(n, a)$ s.t.

if n is prime: $w(n, a) =$ "*prime*"
if n is composite: $w(n, a) =$ "*composite*" with $\Pr \geq \frac{1}{2}$

$w(n, a)$ returns "composite" iff a is a witness to the compositeness of n, meaning that w can prove n is a composite using a. If a is not a witness then "prime" is returned and it is not known whether n is prime.

15.2.2 Adelman-Huang

$w'(n, a)$ s.t.

if n is prime: $w'(n, a) =$ "*prime*" with $\Pr \geq \frac{1}{2}$
if n is composite: $w'(n, a) =$ "*composite*"

This test is the opposite of the Miller-Rabin test. If it ever returns "prime" then we are guaranteed to have a prime. If it returns "composite" then it is unknown whether n is prime.

15.2.3 Algorithm

These two tests can be combined in an algorithm that is always correct:

while(true)
  choose $a \in [1, n]$ uniformly at random
  if $w(n, a) =$ "*composite*" return "composite" STOP.
  if $w'(n, a) =$ "*prime*" return "prime" STOP.

This algorithm is correct because it only stops and gives an answer when a test returns an answer that is guaranteed correct.

This is a Las Vegas algorithm, which is one that does not have an upper bound on running time but it is guaranteed to eventually return the correct answer and stop with a probability of 1. One could modify the loop so it never repeats an a, which would place an exponential upper bound on the loop. Contrast this with quicksort, which has an upper bound on its running time.

No algorithm exists which is not randomized, is polynomial time, and has been proven to be correct. Miller designed an algorithm which is fast, non-randomized, and polynomial in time, but it has not been proven to be correct.

if Generalized Riemann Hypothesis is true
  if n is composite,
    $\exists a \leq O\left(\log^2 n\right)$ s.t. $w(n, a) =$ "*composite*"

Check $\forall a \in [1, O\left(\log^2 n\right)], w(n, a)$. If $w(n, a)$ ever returns "composite" then return "composite", otherwise return "prime".

15.2.4 Prime Number Theorem

$\lim_{n\to\infty} \frac{\pi(n)}{n/\ln n} = 1$.

$\pi(n)$ is the prime distribution function, which specifies the number of primes that are less than or equal to n. Using the prime number theorem we can get, for a randomly chosen number n, $\Pr[n \text{ is prime}] \approx \frac{1}{\ln n}$. This shows that roughly one number in every $\ln n$ is prime.

15.3 Tail Inequalities

Definition: A bound on the probability that a random variable is far from its expectation.

An example of this is quicksort, which has a bound on the expected number of comparisons of $2n \ln n$.

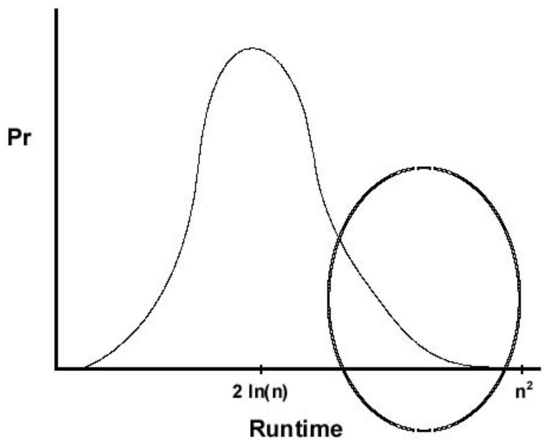


Figure 15.1: Tail Inequality Graph. The circled part of the graph is the tail, where the actual running time is far worse than the expected running time.

15.3.1 Occupancy Problem

Toss m balls independently into n bins. Each ball is equally likely to land in any bin.

Analogy:	1	2
Ball:	data item	job
Bin:	hash table location	processor
Tossing ball:	computing hash address	assign to random processor

Z_i : number of balls in bin i. (Z is a random variable)
Z : $\max_{i=1}^{n}(Z_i)$

Specific case: $m = n$

We want to find a bound for the probability that a particular bin has more than k balls in it.
$\mathrm{E}[Z_i] = \sum_{i=1}^{n} \frac{1}{n} = 1$ (expected number of balls in bin i)
$\Pr[Z_i \geq k] = \Pr[\exists b_1, b_2, \ldots, b_k \text{ s.t. } b_1, \ldots, b_k \text{ land in bin } i]$
$W_{b_1,b_2,\ldots,b_k}$ is the probability of a particular set of balls landing in a particular bin i.
$\Pr[W_{b_1,\ldots,b_k}] = \frac{1}{n^k}$
Choices for $b_1, \ldots, b_k$: $\binom{n}{k}$
$\Pr[Z_i \geq k] \leq \frac{\binom{n}{k}}{n^k} \leq \frac{\left(\frac{ne}{k}\right)^k}{n^k} = \left(\frac{e}{k}\right)^k$ because $\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \left(\frac{ne}{k}\right)^k$.

Exercise:
Show that for $k^* = \left\lceil \frac{2e \ln n}{\ln \ln n} \right\rceil$, $\left(\frac{e}{k^*}\right)^{k^*} < n^{-2}$ and $\Pr[Z \geq k^*] \leq \frac{1}{n}$.
Note that Z is very close to k^* with a high probability.

Theorem 15.1 *Markov's Inequality: Let X be a non-negative random variable with* $\mathrm{E}[X]$*, then* $\Pr[X \geq \alpha] \leq \frac{\mathrm{E}[X]}{\alpha}$.

Proof:

$$\begin{aligned} Y &= X \quad \text{if } X \geq \alpha \\ &= 0 \quad \text{otherwise} \\ Z &= \alpha \quad \text{if } X \geq \alpha \\ &= 0 \quad \text{otherwise} \end{aligned}$$

$\mathrm{E}[X] \geq \mathrm{E}[Y] \geq \mathrm{E}[Z] = \alpha \cdot \Pr[X \geq \alpha]$
$\Pr[X \geq \alpha] \leq \frac{\mathrm{E}[X]}{\alpha}$.

■

This gives us a tail inequality but it uses expectation and not variance of the distribution, thus it is usually too weak.

15.3.2 Chebyshev's Inequality

Theorem 15.2 *Chebyshev's Inequality: Let X be a random variable with* $\mathrm{E}[X] = \mu$ *and* $\mathrm{Var}[X] = \mathrm{E}[(X-\mu)^2] = \sigma^2$*, then* $\Pr[|X - \mu| \geq t] \leq \frac{\sigma^2}{t^2}$.

This gives us a tighter bound (above and below), and it works for any random variable X.

Proof:

Let $Y = (X - \mu)^2$
$\mathrm{E}[Y] = \sigma^2$
$\Pr[Y \geq t^2] \leq \frac{\sigma^2}{t^2}$ (Markov's Inequality)
$Y \geq t^2$ if $|X - \mu| \geq t$
Therefore, $\Pr[|X - \mu| \geq t] \leq \frac{\sigma^2}{t^2}$.

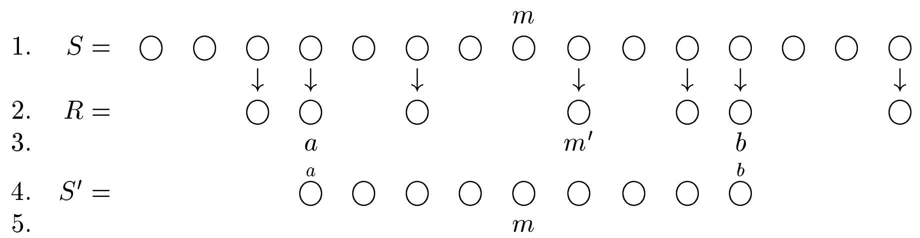
■

15.3.3 Application of Chebyshev's Inequality - Finding the median in $O(n)$ time

Input: an ordered (but not sorted) set S of size n.
Output: the median element m of S.

There are non-random algorithms to solve this but they are complicated and have large constants.

15.3.3.1 Basic Idea



1. Input S is actually unsorted, but in the picture it is sorted from smallest to largest.
2. Let R be a random subset of S.
3. Sort R to get the median m' of R. Then select a and b such that we would guess the median of S lies halfway between them.
4. Let S' be all of the elements in S that fall between a and b.
5. Return m.

15.3.3.2 Algorithm

1. Take a random sample of S of size $n^{\frac{3}{4}}$ with replacement. Call this set R.
2. Sort R $\left(|R| = n^{\frac{3}{4}}\right)$.
3. Find elements a, b such that $rank_R(a) = \frac{1}{2}n^{\frac{3}{4}} - \sqrt{n}$ and $rank_R(b) = \frac{1}{2}n^{\frac{3}{4}} + \sqrt{n}$.

4. Find set $S' = \{x \epsilon S \; s.t. \; a \leq x \leq b\}$.
5. **if** $rank_S(a) \leq \frac{n}{2}$ and $rank_S(b) \geq \frac{n}{2}$ and $rank_S(a) \geq \frac{n}{2} - 2n^{\frac{3}{4}}$ and $rank_S(b) \leq \frac{n}{2} + 2n^{\frac{3}{4}}$.
6. Sort S' and return element m s.t. $rank_{S'}(m) = \frac{n}{2} - rank_S(a)$.
7. Else abort or start over.

Reminder: $rank_R(a) = m$ s.t. a is m^{th} largest element in R.

15.3.3.3 Running Time

1. $O\left(n^{\frac{3}{4}}\right)$
2. $O\left(n^{\frac{3}{4}} \log n^{\frac{3}{4}}\right)$
3. $O(1)$
4. $O(n)$
5. $O(n)$
6. $O\left(n^{\frac{3}{4}} \log n^{\frac{3}{4}}\right)$

$O\left(n^{\frac{3}{4}} \log n^{\frac{3}{4}}\right) \leq O(n)$ because $n^{\epsilon} = \Omega(\log n)$ $(\forall \epsilon > 0)$. This can be shown in greater detail by

$$\begin{aligned} O\left(n^{\frac{3}{4}} \log n^{\frac{3}{4}}\right) &\leq O\left(n^{\frac{3}{4}} \log n\right) && \text{because } O\left(\log n^{\frac{3}{4}}\right) \leq O(\log n) \\ &\leq O\left(n^{\frac{3}{4}} n^{\frac{1}{4}}\right) && \text{because } \log n = o\left(n^{a}\right) \; \forall a > 0 \text{ , so } \log n \leq O\left(n^{\frac{1}{4}}\right) \\ &= O(n). \end{aligned}$$

Therefore, it runs in $O(n)$, and the constants on linear terms are small too.

15.3.3.4 Correctness

If it makes it to step 6 then S' contains the median. In S, $\frac{n}{2} - rank_S(a)$ is the distance between a and m. After sorting S', we can use the distance between a and m to locate and return m in S'. Therefore we must show that step 5 is correct with high probability.

Step 5 is successful if S' contains the median m and $|S'| \leq 4n^{\frac{3}{4}}$. We show that these conditions hold with a high probability:

Theorem 15.3 $\Pr[\textit{algorithm fails}] \leq n^{-\frac{1}{4}}$.

Proof: Algorithm fails iff

I. $rank_S(a) > \frac{n}{2}$ or $rank_S(b) < \frac{n}{2}$
  S' does not contain m.
II. $rank_S(a) < \frac{n}{2} - 2n^{\frac{3}{4}}$ or $rank_S(b) > \frac{n}{2} + 2n^{\frac{3}{4}}$
  $|S'|$ is too large to sort.

I. only holds if there are less than $\frac{1}{2}n^{\frac{3}{4}} - \sqrt{n}$ elements in R left of (smaller than) m, because of the way a and b are chosen in step 3.

Now lets show the probability of I. occurring:

X : number of elements of R left of m.
$X = x_1 + x_2 + x_3 + \ldots + x_{n^{\frac{3}{4}}}$
Where x_i $= 1$ if the i^{th} element chosen is to the left of m
$= 0$ otherwise
$\Pr[x_i = 1] = \frac{1}{2}$
$\mathrm{E}[x_i] = \frac{1}{2}$, $\mathrm{Var}[x_i] = \frac{1}{4}$
$\mathrm{E}[X] = \frac{n^{\frac{3}{4}}}{2}$ (Expectation of sum = sum of expectations)
$\mathrm{Var}[X] = \frac{n^{\frac{3}{4}}}{4}$ (Variance of sum = sum of variances only if variables are independent, which they are in this case)

Thus we can use Chebychev's Inequality:
$\Pr[|X - \frac{n^{\frac{3}{4}}}{2}| \geq \sqrt{n}] \leq \frac{\frac{n^{\frac{3}{4}}}{4}}{(\sqrt{n})^2} = \frac{1}{4n^{\frac{1}{4}}}$.

II. only holds if there are greater than $\frac{1}{2}n^{\frac{3}{4}} - \sqrt{n}$ elements in R left of (smaller than) $\frac{n}{2} - 2n^{\frac{3}{4}}$ in S, because of the way a and b are chosen in step 3.

Now lets show the probability of II. occurring:

Let $rank_S(c) = \frac{n}{2} - 2n^{\frac{3}{4}}$.
X : number of elements of R left of c.
$X = x_1 + x_2 + x_3 + \ldots + x_{n^{\frac{3}{4}}}$
Where x_i $= 1$ if the i^{th} element chosen is to the left of c.
$= 0$ otherwise
$\Pr[x_i = 1] = \frac{\frac{n}{2} - 2n^{\frac{3}{4}}}{n} = \frac{1}{2} - 2n^{-\frac{1}{4}}$
$\mathrm{E}[x_i] = \frac{1}{2} - 2n^{-\frac{1}{4}}$
$\mathrm{Var}[x_i] = \frac{1}{2} - 2n^{-\frac{1}{4}} - [\frac{1}{2} - 2n^{-\frac{1}{4}}]^2 = \frac{1}{4} - 4n^{-\frac{1}{2}}$ $\quad (\mathrm{Var}[x_i] = \mathrm{E}[(x_i - \mu)^2] = \mathrm{E}[x_i^2] - \mathrm{E}[x_i]^2)$
$\mathrm{E}[X] = n^{\frac{3}{4}}\left(\frac{1}{2} - 2n^{-\frac{1}{4}}\right) = \frac{1}{2}n^{\frac{3}{4}} - 2n^{\frac{1}{2}}$ (Expectation of sum = sum of expectations)
$\mathrm{Var}[X] = n^{\frac{3}{4}}\left(\frac{1}{4} - 4n^{-\frac{1}{2}}\right) = \frac{1}{4}n^{\frac{3}{4}} - 4n^{\frac{1}{4}}$ (Variance of sum = sum of variances again in this case)

Thus we can use Chebychev's Inequality:
$\Pr[|X - \frac{1}{2}n^{\frac{3}{4}} - 2n^{\frac{1}{2}}| \geq \sqrt{n}] \leq \frac{\frac{1}{4}n^{\frac{3}{4}} - 4n^{\frac{1}{4}}}{n} = \frac{1}{4n^{\frac{1}{4}}} - \frac{4}{n^{\frac{3}{4}}} \leq \frac{1}{4n^{\frac{1}{4}}}$.

We can now put these probabilities together to get Pr[algorithm fails]:

$$
\begin{aligned}
\Pr[\text{algorithm fails}] &= 1 - \Pr[\text{each of the four tests is false}] \\
&= 1 - (1 - \Pr[\text{individual test is true}])^4 \\
&= 4\Pr[\text{true}] - 6\Pr[\text{true}]^2 + 4\Pr[\text{true}]^3 - \Pr[\text{true}]^4 \\
&\leq \frac{1}{n^{\frac{1}{4}}} - \frac{3}{8n^{\frac{1}{2}}} + \frac{1}{16n^{\frac{3}{4}}} - \frac{1}{256n} \quad (\text{because } \Pr[true] \leq \frac{1}{4n^{\frac{1}{4}}}) \\
&\leq n^{-\frac{1}{4}}.
\end{aligned}
$$

■