Randomized Algorithms

Algorithmics, 186.814, VU 6.0

Günther Raidl

Algorithms and Complexity Group
Institute of Logic and Computation
TU Wien

WS 2022/23, October 4, 2022

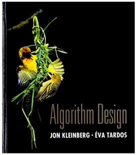
Topics of this part

- Preliminaries
- Randomized primality test
- Basic definitions of probability theory
- Randomized quicksort
- Contention resolution in distributed systems
- Approximation for MAX 3-Satisfiability
- Tail inequalities: Bounding the deviation from the expectation

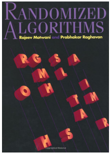
Preliminaries

Literature for this part

- J. Kleinberg, E. Tardos: Algorithm Design, Pearson-Addison Wesley, Chapter 13, 2005



- R. Motwani, P. Raghavan: Randomized Algorithms, Cambridge University Press, 1995



Motivating Example: Quicksort

Idea

Input: Array $A[1 \ldots n]$ of pairwise different elements to be sorted

Divide:
- Select Pivot element $P \in A[1 \ldots n]$
- Reorder (partition) $A[1 \ldots n]$ s.t. $A[1 \ldots n] = (A[1], \ldots, A[p-1], P, A[p+1], \ldots, A[n])$
  - with $A[i] \leq P \; \forall i = 1, \ldots, p-1$, and
  - $A[i] \geq P \; \forall i = p+1, \ldots, n$

Conquer: Recursively sort $A[1 \ldots p-1]$ and $A[p+1 \ldots n]$ as long as there are ≥ 2 elements

Runtime

- Best and average case: $\Theta(n \log n)$
- **Worst case:** $\Theta(n^2)$ (e.g., if array is already sorted)

Randomized Quicksort

- Randomization: Select Pivot element always randomly.

$\rightarrow$ Worst case runtime is still $\Theta(n^2)$, but:
There are no bad input permutations anymore always yielding time $\Theta(n^2)$.

- We are now primarily interested in the **expected runtime**, where the expectation is taken over all possible random decisions.

Theorem (Expected runtime of Randomized Quicksort)

Randomized Quicksort has expected runtime $\Theta(n \log n)$ for all input permutations of A[1. . . n].

Proof to be done.

Randomized Algorithms – General Properties

- Depend on uniformly random numbers as an auxiliary input to guide behavior, usually implemented by a pseudo-random number generator
- Reduce runtime and/or memory for worst case input, avoid dependency on input data
- Can lead to simpler, faster, or only known algorithms for certain problems
- Two variants:

  Monte Carlo algorithms: always time-efficient (e.g. polynom.), but correct output only with high probability

  Las Vegas algorithms: always correct output, but time-efficient only in expectation

Randomized Primality Test

Randomized Primality Test

Given

Positive integer number n (typically very large)

Goal

Determine whether or not n is prime.

Applications

Cryptography (e.g., RSA crypto-system)

Bit complexity

"Large" numbers cannot be added in constant time
$\Rightarrow$ represent n as binary number with $k = \lceil \log_2(n+1) \rceil$ bits,
use number of bit operations as complexity criterion

Division method

Check if n is divisible by 2 or some odd number from $\{3, \dots, \lfloor \sqrt{n} \rfloor\}$ without remainder
Number of divisions $= O(\sqrt{n})$
Corresponding bit complexity: $O(2^{\frac{k}{2}})$ with $k = \Theta(\log n)$
$\Rightarrow$ not applicable in practice!

Miller-Rabin Primality Test

Fermat's Little Theorem

If n is prime, $a^{n-1} \equiv 1 \mod n$ holds for all $a \in \{1, \ldots, n-1\}$.

Corollary

If for some basis a:

- $a^{n-1} \not\equiv 1 \mod n$
  $\Rightarrow n$ is definitely not prime;
  a is called a witness for the compositeness
- $a^{n-1} \equiv 1 \mod n$
  $\Rightarrow n$ is prime with some probability

```
Miller-Rabin Primality Test (n):
for i ← 1,...,s
    if Witness (random (2, n − 1), n)
        return not prime
return prime
```

Idea

If no witness for the compositeness has been found after s iterations, n is likely a prime number.

Is this a Monte Carlo or Las Vegas algorithm?
$\Rightarrow$ Monte Carlo algorithm

Question

How do we efficiently calculate $a^{n-1} \mod n$?

Reformulations

- $a^{2c} \mod n = (a^c \mod n)^2 \mod n$
- $a^{c+1} \mod n = (a^c \mod n) \cdot a \mod n$

Let $(b_{k-1}, b_{k-2}, \ldots, b_0)$ be the number $n-1$ in binary representation.

```
Witness (a, n):
result ← 1        (c ← 0)
for i = k − 1, ..., 0
    result ← (result · result) mod n        (c ← 2c)
    if b_i = 1
        result ← (result · a) mod n        (c ← c + 1)
if result ≠ 1
    return true (a is witness for n not prime)
else
    return false (a is not a witness)
```

Analysis

- Multiplication and modulo of two k-bit numbers: $O(k^2)$
- Witness function: $O(k^3)$
- Total running time of Miller-Rabin algorithm: $O(s \cdot k^3)$

Error probability

If $n > 2$, odd, and not prime:

- There are at least $\frac{n-1}{2}$ witnesses (proof omitted)
- In each iteration, a randomly chosen a is a witness with probability $\geq \frac{1}{2}$
- After s iterations, the probability of not finding a witness is $\leq \frac{1}{2^s}$

Basic Definitions of Probability Theory

Basic Definitions: Finite Probability Space

Sample space Ω

- E.g. all possible outcomes of rolling a dice
- Every point $i \in \Omega$ has a nonnegative probability $p(i)$
- $\sum_{i \in \Omega} p(i) = 1$

Event $\mathcal{E} \subseteq \Omega$

- Probability of $\mathcal{E}$: $\Pr[\mathcal{E}] = \sum_{i \in \mathcal{E}} p(i)$
- E.g. $\mathcal{E} =$ getting an even number, $\Pr[\mathcal{E}] = \frac{1}{2}$
- $\mathrm{Not}(\mathcal{E}) = \Omega - \mathcal{E}$

Conditional Probability, Independence, Union

- Conditional probability of event $\mathcal{E}$ given event $\mathcal{F}$:

$$\Pr[\mathcal{E} \mid \mathcal{F}] = \frac{\Pr[\mathcal{E} \cap \mathcal{F}]}{\Pr[\mathcal{F}]}$$

- Events $\mathcal{E}$ and $\mathcal{F}$ are independent iff $\Pr[\mathcal{E} \mid \mathcal{F}] = \Pr[\mathcal{E}]$; or more generally, events $\mathcal{E}_1, \ldots, \mathcal{E}_n$ are independent iff

$$\Pr\left[\bigcap_{i \in I} \mathcal{E}_i\right] = \prod_{i \in I} \Pr[\mathcal{E}_i] \quad \forall I \subseteq \{1, \ldots, n\}$$

- Union Bound of events $\mathcal{E}_1, \ldots, \mathcal{E}_n$:

$$\Pr\left[\bigcup_{i=1}^{n} \mathcal{E}_i\right] \leq \sum_{i=1}^{n} \Pr[\mathcal{E}_i]$$

Random Variables and Expectations

Random variable $X : \Omega \to \mathbb{N}$
Function from sample space to natural numbers

- We consider **events** $X = j$, $\forall j = 0, \ldots, \infty$
- Corresponding probabilities $\Pr[X = j]$
- Expected value of X: $E[X] = \sum_{j=0}^{\infty} j \cdot \Pr[X = j]$

Example: Waiting for a first success
Suppose some trial is successful with probability $0 < p < 1$
and fails with probability $1 - p$.
X: number of independent trials till first success.

$$E[X] = \sum_{j=0}^{\infty} j \cdot (1-p)^{j-1} p = \frac{p}{1-p} \sum_{j=0}^{\infty} j \cdot (1-p)^{j} = \frac{p}{1-p} \cdot \frac{1-p}{p^2} = \frac{1}{p}.$$

$\rightarrow$ The expected number of independent trials that need to be performed till the first success is $\frac{1}{p}$.

Linearity of Expectation

Let X, Y be random variables over the same probability space.
Then $E[X+Y] = E[X] + E[Y]$.

Example: Guessing cards

- Deck of n cards; repeatedly guess top card before turning over
- X: Number of correctly guessed cards
- $X_i = 1$ iff i-th card is guessed correctly, $X_i = 0$ else.

- Memoryless: $E[X_i] = 0 \cdot \Pr[X_i = 0] + 1 \cdot \Pr[X_i = 1] = \frac{1}{n}$
  $E[X] = \sum_{i=1}^{n} E[X_i] = n \cdot \frac{1}{n} = 1$
- With memory: $E[X_i] = \Pr[X_i = 1] = \frac{1}{n-i+1}$
  $E[X] = \sum_{i=1}^{n} E[X_i] = \sum_{i=1}^{n} \frac{1}{n-i+1} = \sum_{i=1}^{n} \frac{1}{i} = \mathcal{H}_n \approx \ln n$
  with $\mathcal{H}_n$ being called the n**-th Harmonic number**:
  $\ln(n+1) < \mathcal{H}_n \leq 1 + \ln n \qquad \to \mathcal{H}_n \approx \ln n$

Coupon Collector's Problem

- n types of coupons, in each round you get one coupon at random, each type equally likely
- X: number of rounds needed to have one coupon of each type
- **Phase** j: you have already j types and wait for the $(j+1)$-th
- Success probability in one round of phase j: $p(j) = \frac{n-j}{n}$
- X_j: number of rounds in phase j
- $E[X_j] = \frac{1}{p(j)} = \frac{n}{n-j}$
- $E[X] = \sum_{j=0}^{n-1} E[X_j] = n \sum_{j=0}^{n-1} \frac{1}{n-j} = n\,\mathcal{H}_n \approx n \ln n$

Analysis of Randomized Quicksort

Analysis of Randomized Quicksort

Let $A[l \ldots r]$ be the current subarray to be partitioned, and $\text{random}(l, r)$ a function that randomly chooses a value from $\{l, \ldots, r\}$, used to select the Pivot element.

$$\forall i \in \{l, \ldots, r\}: \ \Pr[\text{random}(l, r) = i] = \frac{1}{r - l + 1}.$$

We measure the runtime in terms of number of comparisons $C(n)$, as this is the dominant cost in any reasonable implementation.

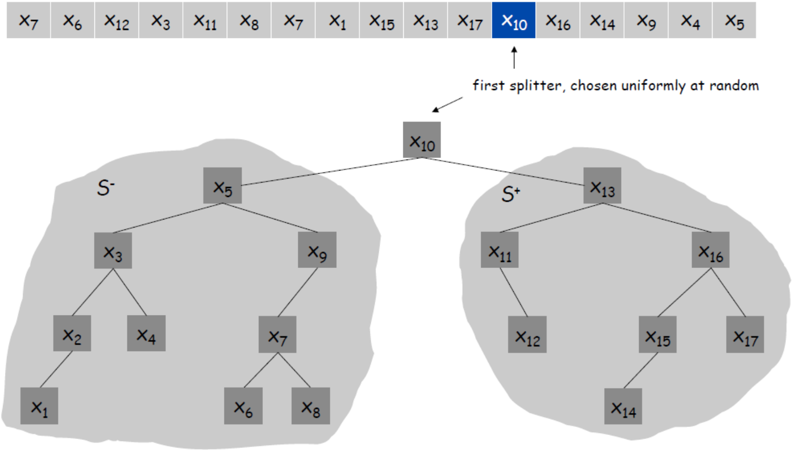
Partitioning according to Pivot element $P \in A[l \ldots r]$:
P is compared once with every other element in $A[l \ldots r]$.

Let $x_1 < x_2 < \ldots < x_n$ be the sorted elements of $A[1 \ldots n]$.
For $i = 1, \ldots, n-1$ and $j = i+1, \ldots, n$ define the indicator variable

$$X_{i,j} = \begin{cases} 1 & \text{if } x_i \text{ is compared to } x_j \\ 0 & \text{otherwise.} \end{cases}$$

Analysis of Randomized Quicksort (cont.)

Recursive call tree can be interpreted as a **binary search tree**, nodes are labeled with chosen Pivot elements:

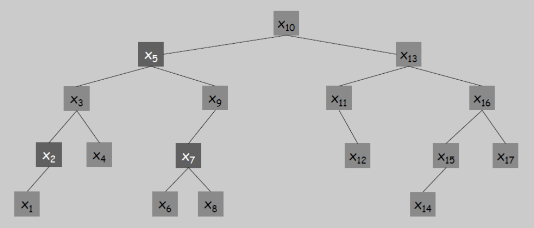


Analysis of Randomized Quicksort (cont.)

Observation:

An Element is only compared with its ancestors and descendants.

- x_2 and x_7 are compared if their *lca* is x_2 or x_7
- x_2 and x_7 are not compared if their *lca* is $x_3, x_4, x_5,$ or x_6



Lemma (Probability of single comparison)

The probability that x_i and x_j are compared is $\Pr[X_{i,j}] = \frac{2}{j-i+1}$.

Analysis of Randomized Quicksort (cont.)

We are interested in the expected total number of comparisons:

$$E[C] = E\left[\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} X_{i,j}\right] = \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} E[X_{i,j}] = \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} \Pr[X_{i,j}] =$$

$$= \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} \frac{2}{\underbrace{j-i+1}_{k}} = \sum_{i=1}^{n-1}\sum_{k=2}^{n-i+1} \frac{2}{k} \le 2\sum_{i=1}^{n}\sum_{k=1}^{n} \frac{1}{k} = 2n\mathcal{H}_n,$$

with $\mathcal{H}_n = \sum_{k=1}^{n} 1/k$ being the n-th Harmonic number:

$$\ln(n+1) < \mathcal{H}_n \le 1 + \ln n \qquad \to \mathcal{H}_n \approx \ln n$$

Thus, $E[C] = E[T] = \Theta(n \log n)$, and we will later show that this expected time is not exceeded with **very high probability**.

Contention Resolution

Contention Resolution in a Distributed System

Contention resolution

- Given n processes $P_1, \ldots, P_n$ competing for access to a shared database (DB).
- If ≥ 2 processes access DB simultaneously, all processes are locked out.
- Devise protocol to ensure all processes get through as frequently as possible.
- **Restriction:** Processes cannot communicate.
- **Challenge:** Symmetry-breaking is needed.

Contention Resolution: Randomized Algorithm

Contention Resolution Algorithm

Each process requests access at each timeslot t
with probability $p = \frac{1}{n}$.

Lemma

Let $S[i,t]$ = event that process i succeeds in accessing DB at time t.
Then $\frac{1}{en} \leq \Pr[S(i,t)] \leq \frac{1}{2n}$, and thus $\Pr[S(i,t)] = \Theta(\frac{1}{n})$.

Because of independence, $\Pr[S(i,t)] = p(1-p)^{n-1}$
(i requests access and all others do not)

$p = \frac{1}{n}$ maximizes $\Pr[S(i,t)]$ $\quad\rightarrow\quad$ $\Pr[S(i,t)] = \frac{1}{n}(1-\frac{1}{n})^{n-1}$

Useful facts: As n increases from 2...

- $(1-\frac{1}{n})^n$ converges monotonically from $\frac{1}{4}$ up to $\frac{1}{e}$
  with e = Euler number,
- $(1-\frac{1}{n})^{n-1}$ converges monotonically from $\frac{1}{2}$ down to $\frac{1}{e}$.

Waiting for Process i

Lemma

The probability that process i fails to access the DB in $\lceil en \rceil$ rounds is $\leq \frac{1}{e}$. After $\lceil en \cdot c \ln n \rceil$ rounds, the probability is $\leq n^{-c}$.

$F[i, t]$ = event that process i fails in rounds $1 \ldots t$.

By independence and previous Lemma: $\Pr[F(i,t)] \leq (1 - \frac{1}{en})^t$

- **Choose $t = \lceil en \rceil$:**
  $\Pr[F(i,t)] \leq \left(1 - \frac{1}{en}\right)^{\lceil en \rceil} \leq \left(1 - \frac{1}{en}\right)^{en} \leq \frac{1}{e}$
- **Choose $t = \lceil en \cdot c \ln n \rceil$:** $\Pr[F(i,t)] \leq \left(\frac{1}{e}\right)^{c \ln n} = n^{-c}$

More generally:

- If $\Pr(F[i, \Theta(n)])$ is bound by a constant
- $\Pr(F[i, \Theta(n \log n)])$ is inversely polynomial in n
  $\rightarrow$ success with **high probability** in $\Theta(n \log n)$ rounds

Waiting for All Processes

Theorem

The probability that ***all*** *processes succeed within* $\lceil 2en \ln n \rceil$ *rounds is at least* $1 - 1/n$.

$F[t]$ = event that ≥ 1 processes fail in rounds $1 \dots t$.

$$\Pr[F[t]] = \Pr\left[\bigcup_{i=1}^{n} F[i,t]\right] \overset{\text{(a)}}{\leq} \sum_{i=1}^{n} \Pr[F[i,t]] \overset{\text{(b)}}{\leq} n\left(1 - \frac{1}{en}\right)^t$$

(a) union bound, (b) result from before

Choosing $t = \lceil 2en \ln n \rceil$:

$$\Pr[F[t]] \leq n \cdot \left(\frac{1}{e}\right)^{2\ln n} = n \cdot n^{-2} = \frac{1}{n}.$$

MAX 3-Satisfiability

MAX 3-Satisfiability

MAX-3SAT: Given a set of clauses $C_1, \ldots, C_k$, each of length 3, over a set of binary variables $X = \{x_1, \ldots, x_n\}$, find a variable assignment satisfying as many clauses as possible.

Example:

$$\begin{aligned} C_1 &= x_2 \vee \overline{x_3} \vee \overline{x_4} \\ C_2 &= x_2 \vee x_3 \vee \overline{x_4} \\ C_3 &= \overline{x_1} \vee x_2 \vee x_4 \\ C_4 &= \overline{x_1} \vee \overline{x_2} \vee x_3 \\ C_5 &= x_1 \vee \overline{x_2} \vee \overline{x_4} \end{aligned}$$

Remark: MAX-3SAT is NP-hard.

Idea

Set each variable independently to true with probability $\frac{1}{2}$ and to false otherwise.

MAX-3SAT: Analysis of Random Assignment

Lemma

Given a MAX-3SAT instance with k clauses, the expected number of clauses satisfied by a random assignment is $\frac{7}{8}k$.

Consider random variable $Z_j = \begin{cases} 1 & \text{if clause } C_j \text{ is satisfied} \\ 0 & \text{otherwise.} \end{cases}$

Number of satisfied clauses **Z** $= \sum_{j=1}^{k} Z_j$

$$E[Z] \overset{\text{(a)}}{=} \sum_{j=1}^{k} E[Z_j] = \sum_{j=1}^{k} \Pr[C_j \text{ is satisfied}] = \frac{7}{8}k$$

(a) linearity of expectation

MAX-3SAT: Lower Bound on Satisfiable Clauses

Corollary

For every instance of MAX-3SAT there is an assignment that satisfies $\geq \frac{7}{8}$ *of all clauses.*

Proof: As $E[Z] = \frac{7}{8}k$, the probability $\Pr[Z \geq \frac{7}{8}k]$ for constructing such an assignment is positive, and consequently such an assignment must exist.

General method

Show the existence of some structure by providing a random construction process that succeeds with positive probability.

MAX-3SAT: Analysis of Random Assignment

Question: Can we turn this idea into a 7/8-approximation algorithm? In general, a random variable may almost always be below its mean.

Lemma

The probability p that a random assignment satisfies $\geq \frac{7}{8}k$ clauses is $\geq \frac{1}{8k}$.

- p_j: probability that exactly j clauses are satisfied; $p = \sum_{j \geq 7k/8} p_j$
- $\frac{7}{8}k = E[Z] = \sum_{j<7k/8} j\, p_j + \sum_{j \geq 7k/8} j\, p_j \;\leq$
  $\frac{7k-1}{8} \sum_{j<7k/8} p_j + k \sum_{j \geq 7k/8} p_j \;=$

  $= \;\frac{7k-1}{8}(1-p) + k\,p \;\leq\; \frac{7k-1}{8} + k\,p$

$\rightarrow \;k\,p \geq \frac{7}{8}k - \frac{7k-1}{8} = \frac{1}{8}, \quad p \geq \frac{1}{8k}$

MAX-3SAT: Johnson's Algorithm

Johnson's Algorithm

Repeatedly generate random assignments until one satisfies $\geq \frac{7}{8}k$ clauses.

Theorem (7/8 Approximation of MAX-3SAT)

Johnson's Algorithm is a 7/8-approximation algorithm.

- By previous lemma, each iteration succeeds with probability $\geq \frac{1}{8k}$.
- By the waiting-time bound, the expected number of trials to find a satisfying assignment is $\leq 8k$.

Total expected runtime: $E[T] = O(8k) \cdot O(n+k) = O(k^2 + kn)$

$\rightarrow$ A Monte Carlo algorithm is turned into a Las Vegas algorithm.

(Hastad, 1997): For MAX-3SAT, no α-approximation algorithm exists for any $\alpha > 7/8$ unless P=NP.

Tail Inequalities: Bounding the deviation from the expectation

(not relevant for exercises and exams, just for your information)

Expected running times are nice, but...

- We might have bad luck and wait "forever"!!

$\rightarrow$ How likely is it we are far off from expectation?

- Three theorems provide bounds on the probability that a random variable is far from its expectation:
    - Markov's inequality
    - Chebyshev's inequalities
    - Chernoff bounds
- Work under different conditions and provide different tightness.

Markov's Inequality

Theorem (Markov's Inequality)

Let X be a non-negativ random variable with expectation $E[X]$. For any $t > 0$

$$\Pr[X \geq t] \leq \frac{E[X]}{t}, \quad \text{or with } t = kE[X] \quad \Pr[X \geq kE[X]] \leq \frac{1}{k}$$

Proof: Let $I_{\geq t}$ be a random variable that is 1 if $X \geq t$ and 0 otherwise.

$$t\,I_{\geq t} \leq X \quad \rightarrow \quad E[t\,I_{\geq t}] \leq E[X] \quad \rightarrow \quad E[I_{\geq t}] \leq \tfrac{E[X]}{t}$$

$$\Pr[X \geq t] = E[I_{\geq t}]$$

$$\rightarrow \quad \Pr[X \geq t] \leq \tfrac{E[X]}{t}$$

Markov's Inequality: Example

n flips of a fair coin; X = number of heads; $\rightarrow\ E[X] = \frac{n}{2}$

$\Pr[X \geq \frac{3}{4}n] \ = \ \Pr[X \geq \frac{3}{2}E[X]] \ \leq \ \frac{2}{3}$

- Tightest possible bound when we only know $E[X]$ and $X \geq 0$.
- Unfortunately often too weak to be useful,
  but provides an important basis.

Chebyshev's Inequality

Variance of X $\sigma^2[X] = E[(X - E[X])^2] = E[X^2] - E[X]^2$
Standard deviation $\sigma[X] = \sqrt{\sigma^2[X]}$

Theorem (Chebyshev's Inequality)

Let X be a non-negativ random variable with expectation $E[X]$ and standard deviation $\sigma[X]$. For any $t > 0$

$$\Pr[|X - E[X]| \geq t\sigma[X]] \leq \frac{1}{t^2}$$

Proof: Random variable $Y = (X - E[X])^2$ has expectation $\sigma^2[X]$.

Using Markov's inequality:

$\Pr[|X - E[X]| \geq t\sigma[X]] \ = \ \Pr[(X - E[X])^2 \geq t^2\sigma^2[X]] \ =$

$= \ \Pr[Y \geq t^2 E[Y]] \leq \frac{1}{t^2}$

Chebyshev's Inequality: Example

n flips of a fair coin; $E[X] = \frac{n}{2}$, $\sigma[X] = \sqrt{\frac{n}{4}}$

More generally, coin flips are Bernoulli trials with $p = \frac{1}{2}$:

- Random variable $Z \in \{0, 1\}$
- $\Pr[Z = 1] = p, \ \Pr[Z = 0] = 1 - p$
- $E[Z] = p, \ \sigma = \sqrt{p(1-p)}$
- Let X = sum of n independent Bernoulli trials with common p.
  - X has the binomial distribution:
  - $\Pr[X = k] = \binom{n}{k} p^k (1-p)^{n-k}$
  - $E[X] = pn, \ \sigma^2 = np(1-p)$

$$\Pr[X - E[X] \geq \tfrac{3}{4}n] + \Pr[X - E[X] \leq \tfrac{n}{4}] \ = \ \Pr[|X - \tfrac{n}{2}| \geq \tfrac{n}{4}] \ \leq \ \frac{1}{t^2}$$

$$t\sigma[X] = \tfrac{n}{4}, \rightarrow \ t = \sqrt{\tfrac{n}{4}}$$

$$\rightarrow \ \Pr[|X - \tfrac{n}{2}| \geq \tfrac{n}{4}] \ \leq \ \frac{4}{n}$$

Chebyshev's Inequality: Example Randomized Quicksort

We have shown: $E[C] = 2n\mathcal{H}_n \approx 2n \ln n$

Knuth (1973): $\sigma[C] \approx 0.65n$

$$\Pr[|C - E[C]| \geq t\sigma[C]] \;\approx\; \Pr[|C - 2n\ln n| \geq t \cdot 0.65n] \;\leq\; \frac{1}{t^2}$$

E.g.: If $n = 10^6$, $\Pr[C \geq 4n\ln n] \leq 0.06\%$

Chernoff Bounds (above mean)

Theorem (Chernoff Bounds (above mean))

Let $X = X_1 + \ldots + X_n$ *be the sum of independent 0–1 random variables.*
For any $\mu \geq E[X]$ *and for any* $\delta > 0$

$$\Pr[X > (1+\delta)\mu] < \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{\mu}$$

It practically means that the sum of independent 0–1 random variables is "tightly centered on the mean; deviations are exponentially unlikely".

Chernoff Bounds (above mean) – Proof

For any $t > 0$,

$$\Pr[X > (1+\delta)\mu] \; = \; \Pr[e^{tX} > e^{t(1+\delta)\mu}] \; \leq \; e^{-t(1+\delta)\mu} \cdot E[e^{tX}]$$

(due to Markov's inequality)

$$E[e^{tX}] \; = \; E[e^{t\sum_{i=1}^{n} X_i}] \; = \; \prod_{i=1}^{n} E[e^{tX_i}]$$

Let $p_i = \Pr[X_i = 1]$. Then

$$E[e^{tX_i}] \; = \; p_i e^t + (1-p_i)e^0 \; = \; 1 + p_i(e^t - 1) \; \leq \; e^{p_i(e^t-1)}$$

($1 + \alpha \leq e^{\alpha}$, $\forall \alpha \geq 0$)

Combining everything:

$$\Pr[X > (1+\delta)\mu] \; \leq \; e^{-t(1+\delta)\mu} \prod_{i=1}^{n} E[e^{tX_i}] \; \leq \; e^{-t(1+\delta)\mu} \prod_{i=1}^{n} e^{p_i(e^t-1)}$$

$$\leq \; e^{-t(1+\delta)\mu} e^{\mu(e^t-1)} \qquad (\sum_{i=1}^{n} p_i = E[X] \leq \mu)$$

Finally, choose $t = \ln(1+\delta)$.

Chernoff Bounds (below mean)

Theorem (Chernoff Bounds (below mean))

Let $X = X_1 + \ldots + X_n$ be the sum of independent 0–1 random variables.
For any $\mu \geq E[X]$ and for any $0 < \delta < 1$

$$\Pr[X < (1-\delta)\mu] < e^{-\delta^2 \mu/2}$$

Proof: Idea similar.

Remark: Not quite symmetric since only makes sense to consider $\delta < 1$.

Chernoff Bounds: Example

n coin flips, X = number of heads

Let $\mu = E[X] = \frac{n}{2}$, $\delta = 0.5$.

$$\Pr[X > (1+0.5)E[X]] = \Pr[X > \frac{3}{4}n] \; < \; \left(\frac{\sqrt{e}}{1.5^{1.5}}\right)^{n/2} \; \approx \; 0.9^{n/2}$$

- $n = 2: \; \Pr[X > 1.5] = 0.25 < 0.9$
- $n = 3: \; \Pr[X > 2.25] = 0.125 < 0.86$
- $n = 4: \; \Pr[X > 3] = 0.0625 < 0.81$
- $n = 100: \; \Pr[X > 75] =? < 0.0052$

$\rightarrow$ Chernoff bounds still overestimate significantly but are easy