CS 473  Algorithms: Lecture 7

- randomized algo: - O(k) w.p. 1/k
- can use on actual computer
- dynamic programming, optimized
- can use in practice
- longest increasing subsequence
- probability review
- randomized algo
- quicksort

Q: what is the "most realistic" model of computation? - captures abilities of actual computer

Q: how does this help? X - returns 0 ≤ i ≤ k w.p. 1/k
- random dimension II

Q: who will win the next US election? - "computational" problem II
- sum of numbers I
- 2.35 million eligible voters
- 1.38 million actual voters
- 1.38 million actual voters
- estimate actual win in 55%
- error II
- is actually related to algo II
- I can prove I am correct II
- Chernoff II

Q: why rand algo?
- can be simpler, but analysis can be more complex
- faster, but may fail with some probability
- sometimes only known efficient algo is randomized, e.g. primality testing

Q: what is randomness?

def: P: finite/countable domain, Pr: P \rightarrow [0,1], then (P, Pr) is
- a discrete probability space if \sum_{w \in P} Pr[w] = 1.

Def: can define general probability space, e.g. \mathbb{R} = \mathbb{R}, but it tricky
- discrete space; maybe here

\text{eg. } R = \{H, T\} \text{, } Pr[H] = 1/2 \text{, } Pr[T] = 1/2 \text{, two fair dice}
- \text{eg. } R = \{0, 1, 2\} \text{, } \min_{\in \mathbb{R}} 3, Pr[1] = \text{probability algo takes } i \text{ steps}

Def: a probability space. An event A \subseteq R, P[A] = \sum_{w \in A} Pr[w]
- A = \{H, T\} \text{, first win is heads}
- A = \{T\} \text{, second win is tails}
- \overline{A} = R \setminus A
- A \cap B, A \cup B
def: $(\Omega, P)$ prob. space. $A, B$ events. $A$ and $B$ are independent

e.g.: $\Omega = \{ H, T \}$, $A = \{ H \}$, $B = \{ TT, HT \}$

$P[A] = \frac{1}{2}$, $P[B] = \frac{1}{2}$,
$P[A \cap B] = P[HT] = \frac{1}{4}$

$P[A \cap B] = P[A]P[B] \Rightarrow \frac{1}{4} = \frac{1}{2} \times \frac{1}{2}$

def: $(\Sigma, P)$ prob. space. $A$ & $B$ events. Define the conditional probability of $A \mid B$ to be
$P[A \mid B] = \frac{P[A \cap B]}{P[B]}$ where $P[B] \neq 0$.

lem: $(\Sigma, P)$ prob. space. $A, B$ events, $P[B] \neq 0$.
Then
- $(B, P[B])$ is a probability space
- $P[A \cap B] = P[A \mid B] \cdot P[B]
- A, B independent if $P[A \mid B] = P[A]$

Pr: exercise


Pr: \[ \bigcup \]

\[ \text{simple, but powerful!} \]

def: $(\mathcal{D}, P)$ A random variable is $X: \mathcal{D} \rightarrow \mathbb{R}$

$X: \mathcal{D} \rightarrow \mathbb{R}$, $Y: \mathcal{D} \rightarrow \mathbb{R}$ are independent if $f_X \cdot f_Y$.

E.g.: if $X = \omega$ and $Y = \beta$, are independent

equv: $P[X = \omega \land Y = \beta] = P[X = \omega] \cdot P[Y = \beta]$

e.g.: $\text{R} = \{ HH, HT, TH, TT \}$, $P[HT] = \frac{1}{4}$

$X = 1$ if 1st coin
$Y = 0$ if second coin
$P[X = Y] = \frac{1}{2}$

A & D are independent

Pr: $(\mathcal{R}, P)$ $X$ random var. The expectation of $X$ is $E[X] = \sum_{\omega \in \Omega} X(\omega)P[\omega]$

The conditional expectation if $X$ conditioned on $A$ is $E[X \mid A] = \sum_{\omega \in \Omega} X(\omega) \cdot P[\omega \mid A]$

lem: $X, Y$ independent $\Rightarrow E[XY] = E[X] \cdot E[Y]$ Exercise I

lem: $X, Y$ independent $\Rightarrow \mathbb{E}[XY] - \mathbb{E}[X] \cdot \mathbb{E}[Y]$ Exercise II

def: if $X$ random var. if $X: \mathcal{D} \rightarrow \{0, 1\}$ is a binary random variable

if $A \in \mathcal{D}$ and $X(\omega) = 1$ for $A$ then $X$ is an indicator random variable

lem: $X = 1$ if $A$. Then $E[X] = P[A]$

prop: to compute $E[X]$ should union $X = \bigcup_{A \in \mathcal{D}}$ $= \sum_{A \in \mathcal{D}} P[A]$
Q: how to model randomized algo?

models: deterministic, randomized algo f

- worst case input x
  \[ x \mapsto f(x) \]

- complexity: \[ T(n) = \max_{|x| \leq n} T(x) \]

- probability: deterministic f
  \[ x \sim \mathcal{D}_n \quad \text{input distribution} \]
  \[ x \mapsto f(x) \]

- complexity: \[ T(n) = \max_{|x| \leq n} \mathbb{E}[T(x)] \]

- randomized algo: randomized algo f \[ \text{via } \text{rand}(.), \text{rand}(.). \]

- worst case input x \[ x \mapsto f(x) \]

- complexity: \[ T(n) = \max_{|x| \leq n} \mathbb{E}[T(x)] \]

- randomized vs deterministic

- 

\[ \text{def: } \text{err (error): } E(x) = \Pr[f(x) \neq f(x)] \text{ in case } T \]

- worst case: \[ E(n) = \max_{|x| \leq n} E(x) \]

\[ \text{def: } A \text{ las vegas algo } \quad T(n) \leq n \quad \text{if no error} \]

\[ \text{def: } A \text{ monte carlo algo } \quad \forall x \quad T(x) \leq \log_2 \frac{\epsilon}{\delta} \quad \text{always fast} \]

- \[ E(n) \text{ is "small" } \leq \frac{\epsilon}{\delta} \quad \text{\# okay} \]

\[ \text{rank: } \text{Las Vegas is "almost" deterministic } \leq \frac{n}{\log n} \text{ pretty good } \]

\[ \text{real error comes from } E(n) > 0 \leq \frac{\epsilon}{\delta} \text{ essentially always risk of overkill} \]

\[ \text{fact: any efficient randomized algo is wrong Monte Carlo } \text{ it's all about the error} \]

\[ \text{Q: example?} \]

\[ \text{quickSort}(a_1, a_2, \ldots, a_n) = \tilde{a} \quad \text{sort } \tilde{a} \]

\[ \text{pick pivot } a_{i_0} \quad \text{somewhere} \tilde{a} \]

\[ \text{partition } \tilde{a} = (b_1, a_{i_0}, b_2) \text{ w/ } b_1 < a_{i_0} < b_2 \]

\[ \text{recur } \tilde{a} = (\text{quickSort}(\tilde{b}_1), a_{i_0}, \text{quickSort}(\tilde{b}_2)) \]

\[ \text{correctness: clear } \]
complexity:
\[ j = \text{rank}(g_{ij}) \implies |E| = j - 1 \implies \mathbb{E}[n] = n - j \]
\[ T(n) \leq O(n) + T(n-j) + T(n-i) \]
\[ \leq \ldots \leq O(n^2) \]

and:
\[ T(n) \geq \Omega(n^2) \]

**Main Idea:** Choose \( g_{i0} \) so \( \text{rank}(g_{i0}) \leq \sqrt{n} \) \( \iff \text{line a balanced 1/2} \)

- **pick** \( g_{i0} \) via \( i_0 = \text{rand}(n) + 1 \in \{1, \ldots, n\} \) \( \iff \text{1 rand} \)

**Future**

\[ T(n) \leq T\left(\frac{\sqrt{n}}{2}\right) + T(n_0) + O(n) \text{ \ with probability } 2/3 \]
\[ \leq \ldots \leq O(n \log n) \text{ \ not a formal proof} \]

**Fact:** can select \( g_{i0} \) \( \iff \text{rand}(n) \leq \sqrt{n} \) \( \iff \text{O(n log n)} \)

**Remark:** \( = O(n \log n) \text{ deterministic quicksort} \)

- but is not as efficient/simply as randomized case.

This: randomized quicksort \( \iff \text{rand}(n) = i_0 \)

- error probability \( O(1) \text{ \ when more} \)

- for all \( a, \ E[T(a, -a)] \leq O(n \log n) \text{ \ when} \)

**Proof:**
\[ T(\bar{E}) = \text{compensation} \]

- main bulk of work \( \iff \text{more work} \)

\[ J = \text{all calls to rand(\cdot)} \text{ \ I complicate \ 2 \ \ I how to decouple? \ 2} \]
\[ \bar{a} \implies (\bar{E}, g_{i0}, \bar{E}) \]

\[ E[T(\bar{E})] \leq O(n) + E[T(\bar{E})] + \sum_{i=1}^n \frac{1}{n} \]

\[ \leq O(n) + \sum_{i=1}^n T(\bar{E}) \text{ \ rank}(g_{i0}) = i \text{ \ rank}(g_{i0}) \leq i \sum_{i=1}^n \frac{1}{n} \leq \frac{1}{n} \]

\[ = O(n \log n) \]

\[ = 0(n) + \frac{1}{n} \sum_{i=1}^n (T(i-1) + T(n-i)) \leq \text{any} \bar{a} \]

\[ = T(n) \leq O(n) + \frac{1}{n} \sum_{i=1}^n (T(i-1) + T(n-i)) \]

\[ \leq \ldots \leq \text{prob.} O(n \log n) \text{ \ rand algo}\]

**QuickSort**
- non-trivial bands