

CSE 203A: Randomized Algorithms

Spring 2026

Lecture 3 Adelman's Theorem and Linearity of Expectation

Date: 04/03/26

Instructor: Daniel M. Kane

Scribe: Kyle Lee

1 Recap

Complexity classes

- PP (Probabilistic Polynomial)
 - $x \in L \implies P(YES) > \frac{1}{2}$
 - $x \notin L \implies P(NO) < \frac{1}{2}$
- BPP (Bounded Probabilistic Polynomial)
 - $x \in L \implies P(YES) \geq \frac{2}{3}$
 - $x \notin L \implies P(NO) \leq \frac{1}{3}$
 - Gap between probability > 0
 - Bound is arbitrary, can improve by rerunning algorithm multiple times
 - Always polynomial time
- ZPP (Zero-error Probabilistic Polynomial)
 - Always correctly return YES or NO
 - Expected value of the run time is polynomial
- RP (Randomized Polynomial)
 - $x \in L \implies P(YES) \geq \frac{1}{2}$
 - $x \notin L \implies P(NO) = 1$

2 Adelman's Theorem

In particular, we can improve probability of failure of BPP algorithm to 2^{-2n}

- Run it on linear number of strings
- Pick $m = O(n)$ random strings r_1, \dots, r_m
- Run $A(x, r_1), A(x, r_2), \dots, A(x, r_m)$ and take majority result

Since, 2^{-2n} is small and the number of inputs of size n is only 2^n , the expected number of inputs for which we have wrong answer $\leq 2^{-n} < 1$. For some choice of r_1, \dots, r_m , the algorithm never fails

Theorem 2.1 (Adelman's Theorem).

$$BPP \subset P/poly$$

$P/poly$ is the class P , but given polynomially many bits of advice based on input size. It is also the class of irregular poly-sized circuits. In this case, we can think of the random strings r_1, \dots, r_m as the advice.

There exists a correct set of random strings that give you the right outcome for any input on that size. Explicitly finding these random strings aren't easy. Almost all set of strings have this property, but an explicit set that you can prove is often not easy. This is related to derandomization.

Question: Does Randomness help?

- Adelman's Theorem - for irregular circuits the answer is no ($BPP/poly = P/poly$)
- Restricted computational models - yes
- For regular circuit families - maybe (open question whether $P = BPP$, most complexity theorist have the idea that they're probably equal)

Theorem 2.2. *If $NP \subset BPP$, then $NP = RP$.*

Proof. First show $RP \subset NP$ unequivocally. Let $A(x, r)$ be a RP algorithm such that

1. If $x \in L$, then $A(x, r) = 1$ for at least 50% of r 's.
2. If $x \notin L$, then $A(x, r) = 0$ for all r 's

Let r be the advice/hint given to an NP algorithm to verify in polytime. For the algorithm to be in NP , there only needs to exist one advice/hint r (50% is at least 1) such that $A(x, r)$ gives the correct answer. Thus, $RP \subset NP$.

We still need to show the other direction $RP \supset NP$. If $NP \subset BPP$, then for any NP problem, there exists a poly time randomized algorithm that with $> 50\%$ probability finds a satisfying assignment if one exists. For each bit in an assignment, run BPP algorithm to find all bits. To get RP algorithm, verify this assignment using the witness generated by BPP . This has the same properties of an RP algorithm. \square

3 Probability tools (Ch 3 & 4)

3.1 Linearity of Expectation

Theorem 3.1 (Linearity of Expectation). *For any random variables x, y*

$$\mathbb{E}[x + y] = \mathbb{E}[x] + \mathbb{E}[y]$$

3.1.1 Example 1: Expected number of local maxima of random permutation of $\{1, \dots, n\}$

Expected number of local maxima of random permutation of $\{1, \dots, n\}$, e.g. for $[2, 1, 3]$, the local maxima are 2 and 3. The indicator random variable of an event E is defined as:

$$\mathbb{1}_E = \begin{cases} 1 & \text{if } E \text{ is true} \\ 0 & \text{otherwise} \end{cases}$$

The expected value is then

$$\mathbb{E}[\mathbb{1}_E] = 1 \cdot P(E) + 0 \cdot P(E^c) = P(E)$$

The number of local maxima of random permutation of $\{1, \dots, n\}$ can be rewritten as a summation of indicator random variables

$$\sum_{i=1}^n \mathbb{1}_{i \text{ is a local max}}$$

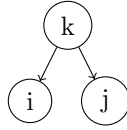
Then, it becomes easier to write the expectation use linearity.

$$\begin{aligned} \mathbb{E} \left[\sum_{i=1}^n \mathbb{1}_{i \text{ is a local max}} \right] &= \sum_{i=1}^n \mathbb{E}[\mathbb{1}_{i \text{ is a local max}}] \\ &= \sum_{i=1}^n P(i \text{ is local max}) \\ &= \frac{n-2}{3} + \frac{1}{2} + \frac{1}{2} \end{aligned}$$

To derive the last line, we know that $P(i \text{ is local max}) = 1/3$ unless $i = 1$ or n (then its $1/2$).

3.1.2 Example 2: Expected number of randomized quick sort comparisons

Can derive a random binary tree (treap). Each x in the tree has to be compared to all of its descendants. We first assign each element a random priority in $[0, 1]$. Pick the pivot as the number with lowest priority. When is i compared to j ? If some other pivot k between them is selected first, then i is not compared to j . We can rephrase the question to ask if there exists a k such that $i < k < j$ with $\text{priority}(k) < \min(\text{priority}(i), \text{priority}(j))$.



Again, using the indicator random variable, we can rewrite the expectation

$$\begin{aligned}
 & \mathbb{E}[\text{number of quicksort comparisons of sorting } \{1, \dots, n\}] \\
 &= \mathbb{E} \left[\sum_{1 \leq i \leq j \leq n} \mathbb{1}_{i \text{ compared to } j} \right] \\
 &= \sum_{1 \leq i \leq j \leq n} \mathbb{E}[\mathbb{1}_{i \text{ compared to } j}] \\
 &= \sum_{1 \leq i \leq j \leq n} P(i \text{ is compared to } j) \\
 &= \sum_{1 \leq i \leq j \leq n} P(\text{no } k \text{ between } i \text{ and } j \text{ with smaller priority}) \\
 &= \sum_{1 \leq i \leq j \leq n} \frac{2}{j - i + 1}
 \end{aligned}$$

Rewrite the summation using $j - i + 1 = d$

$$\begin{aligned}
 \sum_{1 \leq i \leq j \leq n} \frac{2}{j - i + 1} &= \sum_{d=2}^n (n - d + 1) \left(\frac{2}{d}\right) \\
 &= 2(n - 1) \sum_{d=2}^n \frac{1}{d} - \sum_{d=2}^n 2 - 2(n - 1) \\
 &= 2nH_n - 4n - 2H_n + 4
 \end{aligned}$$

where H_n is the n -th harmonic number.