Lecture 3: Probabilistic Analysis, Randomized Quicksort

Michael Dinitz

September 8, 2020
601.433/633 Introduction to Algorithms
Introduction: Sorting

- Sorting: given array of comparable elements, put them in sorted order
- Popular topic to cover in Algorithms courses
- This course:
  - I assume you know the basics (mergesort, quicksort, insertion sort, selection sort, bubble sort, etc.) from Data Structures
  - Today: more advanced sorting (randomized quicksort)
  - Next week: Sorting lower bound and ways around it.
First lecture: “Average-case” problematic.
- What is the “average case”?
- Want to design algorithms that work in all applications.
Randomized Algorithms and Probabilistic Analysis

First lecture: “Average-case” problematic.
- What is the “average case”?  
- Want to design algorithms that work in *all* applications.

Instead of assuming random distribution over inputs (average-case analysis, machine learning), add randomization *inside* algorithm!
- Still assume worst-case inputs, give bound on worst-case *expected* running time.
Randomized Algorithms and Probabilistic Analysis

First lecture: “Average-case” problematic.
  ▶ What is the “average case”?
  ▶ Want to design algorithms that work in all applications.

Instead of assuming random distribution over inputs (average-case analysis, machine learning), add randomization inside algorithm!
  ▶ Still assume worst-case inputs, give bound on worst-case expected running time.

Fall semesters: 601.434/634 Randomized and Big Data Algorithms. Great class!
Randomized Algorithms and Probabilistic Analysis

First lecture: “Average-case” problematic.
  ▶ What is the “average case”?
  ▶ Want to design algorithms that work in all applications.

Instead of assuming random distribution over inputs (average-case analysis, machine learning), add randomization inside algorithm!
  ▶ Still assume worst-case inputs, give bound on worst-case expected running time.

Fall semesters: 601.434/634 Randomized and Big Data Algorithms. Great class!

Today: adding randomness into quicksort.
Quicksort Basics (Review)

Input: array $A$ of length $n$.

Algorithm:
1. If $n = 0$ or $1$, return $A$ (already sorted)
2. Pick some element $p$ as the pivot
3. Compare every element of $A$ to $p$. Let $L$ be the elements less than $p$, let $G$ be the elements larger than $p$. Create array $[L, p, G]$
4. Recursively sort $L$ and $G$. 
QuickSort Basics (Review)

Input: array $A$ of length $n$.

Algorithm:
1. If $n = 0$ or $1$, return $A$ (already sorted)
2. Pick some element $p$ as the pivot
3. Compare every element of $A$ to $p$. Let $L$ be the elements less than $p$, let $G$ be the elements larger than $p$. Create array $[L, p, G]$
4. Recursively sort $L$ and $G$.

Not fully specified: how to choose $p$?
- Traditionally: some simple deterministic choice (first element, last element, etc.)
- Next lecture: better deterministic choice (not very practical)
- Now: first element
Quicksort Analysis

**Upper bound:**
If \( p \) picked as pivot in step 2, then in correct place after step 3.
Quicksort Analysis

Upper bound:
If \( p \) picked as pivot in step 2, then in correct place after step 3
\[ \rightarrow \text{ step 2 and 3 executed at most } n \text{ times.} \]
Quicksort Analysis

Upper bound:
If p picked as pivot in step 2, then in correct place after step 3
implies step 2 and 3 executed at most n times.

Step 3 takes time \( O(n) \) (compare all to pivot)
Quicksort Analysis

**Upper bound:**
If p picked as pivot in step 2, then in correct place after step 3

⇒ step 2 and 3 executed at most n times.

Step 3 takes time $O(n)$

⇒ total time at most $O(n^2)$
QuickSort Analysis

**Upper bound:**
If $p$ picked as pivot in step 2, then in correct place after step 3
$\implies$ step 2 and 3 executed at most $n$ times.

Step 3 takes time $O(n)$
$\implies$ total time at most $O(n^2)$

**Lower Bound:**
Suppose $A$ already sorted.
Quicksort Analysis

**Upper bound:**
If $p$ picked as pivot in step 2, then in correct place after step 3
$\implies$ step 2 and 3 executed at most $n$ times.

Step 3 takes time $O(n)$
$\implies$ total time at most $O(n^2)$

**Lower Bound:**
Suppose $A$ already sorted.
$\implies p = A[0]$ is smallest element
Quicksort Analysis

**Upper bound:**
If \( p \) picked as pivot in step 2, then in correct place after step 3
\[ \implies \text{step 2 and 3 executed at most } n \text{ times.} \]

Step 3 takes time \( O(n) \)
\[ \implies \text{total time at most } O(n^2) \]

**Lower Bound:**
Suppose \( A \) already sorted.
\[ \implies p = A[0] \text{ is smallest element} \implies L = \emptyset \text{ and } G = A[1..n-1] \]
QuickSort Analysis

**Upper bound:**
If p picked as pivot in step 2, then in correct place after step 3
\[\implies\] step 2 and 3 executed at most \(n\) times.

Step 3 takes time \(O(n)\)
\[\implies\] total time at most \(O(n^2)\)

**Lower Bound:**
Suppose \(A\) already sorted.
\[\implies\] \(p = A[0]\) is smallest element \[\implies\] \(L = \emptyset\) and \(G = A[1..n-1]\)
\[\implies\] in one call to quicksort, do \(\Omega(n)\) work to compare everything to \(p\), then recurse on array of size \(n-1\)
Quicksort Analysis

**Upper bound:**
If $p$ picked as pivot in step 2, then in correct place after step 3
$\implies$ step 2 and 3 executed at most $n$ times.

Step 3 takes time $O(n)$
$\implies$ total time at most $O(n^2)$

**Lower Bound:**
Suppose $A$ already sorted.
$\implies p = A[0]$ is smallest element $\implies L = \emptyset$ and $G = A[1..n-1]$
$\implies$ in one call to quicksort, do $\Omega(n)$ work to compare everything to $p$, then recurse on array of size $n-1$
$\implies$ running time is $T(n) = T(n-1) + cn$
Quicksort Analysis

Upper bound:
If $p$ picked as pivot in step 2, then in correct place after step 3
$\implies$ step 2 and 3 executed at most $n$ times.

Step 3 takes time $O(n)$
$\implies$ total time at most $O(n^2)$

Lower Bound:
Suppose $A$ already sorted.
$\implies$ $p = A[0]$ is smallest element $\implies L = \emptyset$ and $G = A[1..n-1]$
$\implies$ in one call to quicksort, do $\Omega(n)$ work to compare everything to $p$, then recurse on array of size $n-1$
$\implies$ running time is $T(n) = T(n-1) + cn \implies T(n) = \Theta(n^2)$
Randomized Quicksort

Randomized Quicksort: pick \( p \) uniformly at random from \( A \).

Today: prove that expected running time at most \( O(n \log n) \) for every input \( A \).
Randomized Quicksort

Randomized Quicksort: pick \( p \) uniformly at random from \( A \).

Today: prove that expected running time at most \( \mathcal{O}(n \log n) \) for every input \( A \).

- Better than an average-case bound: holds for every single input!
- Maybe in one application inputs tend to be pretty well-sorted: original deterministic quicksort bad, this still good!
Randomized Quicksort

Randomized Quicksort: pick \( p \) uniformly at random from \( A \).

Today: prove that expected running time at most \( \mathcal{O}(n \log n) \) for every input \( A \).

- Better than an average-case bound: holds for every single input!
- Maybe in one application inputs tend to be pretty well-sorted: original deterministic quicksort bad, this still good!
- Today only expectation. Can be more clever to get high probability bounds.
Randomized Quicksort

Randomized Quicksort: pick $p$ uniformly at random from $A$.

Today: prove that expected running time at most $O(n \log n)$ for every input $A$.

- Better than an average-case bound: holds for every single input!
- Maybe in one application inputs tend to be pretty well-sorted: original deterministic quicksort bad, this still good!
- Today only expectation. Can be more clever to get high probability bounds.

Before doing analysis, quick review of basic probability theory.
Probability Basics I

Only semi-formal here. Look at CLRS Chapter 5, take Introduction to Probability
Probability Basics I

Only semi-formal here. Look at CLRS Chapter 5, take Introduction to Probability

Ω: Sample space. Set of all possible outcomes.
Probability Basics I

Only semi-formal here. Look at CLRS Chapter 5, take Introduction to Probability

Ω: Sample space. Set of all possible outcomes.
   - Roll two dice. Ω =
Probability Basics I

Only semi-formal here. Look at CLRS Chapter 5, take Introduction to Probability

\(\Omega\): Sample space. Set of all possible outcomes.

- Roll two dice. \(\Omega = \{1, 2, \ldots, 6\} \times \{1, 2, \ldots, 6\}\).
Probability Basics I

Only semi-formal here. Look at CLRS Chapter 5, take Introduction to Probability

\( \Omega \): Sample space. Set of all possible outcomes.
- Roll two dice. \( \Omega = \{1, 2, \ldots, 6\} \times \{1, 2, \ldots, 6\}. \) Not \( \{2, 3, \ldots, 12\} \)
Probability Basics I

Only semi-formal here. Look at CLRS Chapter 5, take Introduction to Probability

\( \Omega \): Sample space. Set of all possible outcomes.

- Roll two dice. \( \Omega = \{1, 2, \ldots, 6\} \times \{1, 2, \ldots, 6\} \). Not \( \{2, 3, \ldots, 12\} \)

Event: subset of \( \Omega \)
Probability Basics I

Only semi-formal here. Look at CLRS Chapter 5, take Introduction to Probability

\( \Omega \): Sample space. Set of all possible outcomes.

- Roll two dice. \( \Omega = \{1, 2, \ldots, 6\} \times \{1, 2, \ldots, 6\} \). Not \( \{2, 3, \ldots, 12\} \)

Event: subset of \( \Omega \)

- “Event that first die is 3”: \( \{(3, x) : x \in \{1, 2, \ldots, 6\}\} \)
- “Event that dice add up to 7 or 11”: \( \{(x, y) \in \Omega : (x + y = 7) \text{ or } (x + y = 11)\} \)
Probability Basics I

Only semi-formal here. Look at CLRS Chapter 5, take Introduction to Probability

\( \Omega \): Sample space. Set of all possible outcomes.

- Roll two dice. \( \Omega = \{1, 2, \ldots, 6\} \times \{1, 2, \ldots, 6\} \). \( \text{Not} \ \{2, 3, \ldots, 12\} \)

Event: subset of \( \Omega \)

- “Event that first die is 3”: \( \{(3, x) : x \in \{1, 2, \ldots, 6\}\} \)
- “Event that dice add up to 7 or 11”: \( \{(x, y) \in \Omega : (x + y = 7) \text{ or } (x + y = 11)\} \)

Random Variable: \( X : \Omega \to \mathbb{R} \)

- \( X_1 \): value of first die. \( X_1(x, y) = x \)
- \( X_2 \): value of second die. \( X_2(x, y) = y \)
- \( X = X_1 + X_2 \): sum of the dice. \( X(x, y) = x + y = X_1(x, y) + X_2(x, y) \)
Probability Basics I

Only semi-formal here. Look at CLRS Chapter 5, take Introduction to Probability

\( \Omega \): Sample space. Set of all possible outcomes.

- Roll two dice. \( \Omega = \{1, 2, \ldots, 6\} \times \{1, 2, \ldots, 6\} \). Not \( \{2, 3, \ldots, 12\} \)

Event: subset of \( \Omega \)

- "Event that first die is 3": \( \{ (3, x) : x \in \{1, 2, \ldots, 6\} \} \)
- "Event that dice add up to 7 or 11": \( \{ (x, y) \in \Omega : (x + y = 7) \text{ or } (x + y = 11) \} \)

Random Variable: \( X : \Omega \rightarrow \mathbb{R} \)

- \( X_1 \): value of first die. \( X_1(x, y) = x \)
- \( X_2 \): value of second die. \( X_2(x, y) = y \)
- \( X = X_1 + X_2 \): sum of the dice. \( X(x, y) = x + y = X_1(x, y) + X_2(x, y) \)

Random variables super important! Running time of randomized quicksort is a random variable.
Probability Basics II

Want to define probabilities. Should use measure theory. Won’t.
Want to define probabilities. Should use measure theory. Won’t.

For each $e \in \Omega$ let $\Pr[e]$ be probability of $e$ (probability distribution)

- $\Pr[e] \geq 0$ for all $e \in \Omega$, and $\sum_{e \in \Omega} \Pr[e] = 1$
- Probability of an event $A$ is $\Pr[A] = \sum_{e \in A} \Pr[e]$
Want to define probabilities. Should use measure theory. Won’t.

For each $e \in \Omega$ let $\Pr[e]$ be probability of $e$ (probability distribution)

- $\Pr[e] \geq 0$ for all $e \in \Omega$, and $\sum_{e \in \Omega} \Pr[e] = 1$
- Probability of an event $A$ is $\Pr[A] = \sum_{e \in A} \Pr[e]$

Conditional probability: if $A$ and $B$ are events:

$$Pr[B|A] = \frac{Pr[A \cap B]}{Pr[A]} = \frac{\sum_{e \in A \cap B} Pr[e]}{\sum_{e \in A} Pr[e]}$$
Probability Basics III: Expectations

Expectation of a random variable:

\[ E[X] = \sum_{e \in \Omega} X(e) \Pr[e] \]

“Average” of the random variable according to probability distribution
Probability Basics III: Expectations

Expectation of a random variable:

\[ E[X] = \sum_{e \in \Omega} X(e)Pr[e] \]

“Average” of the random variable according to probability distribution

Can be useful to rearrange terms to get different equation:

\[ E[X] = \sum_{e \in \Omega} X(e)Pr[e] = \sum_{y \in \mathbb{R}} \sum_{e \in \Omega : X(e) = y} y \cdot Pr[e] = \sum_{y \in \mathbb{R}} y \cdot Pr[X = y] \]
Probability Basics III: Expectations

Expectation of a random variable:

$$E[X] = \sum_{e \in \Omega} X(e)Pr[e]$$

“Average” of the random variable according to probability distribution

Can be useful to rearrange terms to get different equation:

$$E[X] = \sum_{e \in \Omega} X(e)Pr[e] = \sum_{y \in \mathbb{R}} \sum_{e \in \Omega : X(e)=y} y \cdot Pr[e] = \sum_{y \in \mathbb{R}} y \cdot Pr[X = y]$$

Conditional Expectation: \( A \) an event, \( X \) a random variable.

$$E[X|A] = \frac{1}{Pr[A]} \sum_{e \in A} X(e)Pr[e]$$
Linearity of Expectations

Amazing feature of expectations: linearity!

Theorem

For any two random variables $X$ and $Y$, and any constants $\alpha$ and $\beta$:

$$E[\alpha X + \beta Y] = \alpha E[X] + \beta E[Y]$$
Linearity of Expectations

Amazing feature of expectations: linearity!

Theorem

For any two random variables $X$ and $Y$, and any constants $\alpha$ and $\beta$:

$$E[\alpha X + \beta Y] = \alpha E[X] + \beta E[Y]$$

Consider rolling two dice. Let $X$ be sum. What is $E[X]$?

- $E[X] = \sum_{e \in \Omega} X(e) \Pr[e]$. 36 term sum!
- $E[X] = \sum_{y \in \mathbb{R}} y \cdot \Pr[X = y]$. What is $\Pr[X = 2]$, $\Pr[X = 3]$, ...?
Linearity of Expectations

Amazing feature of expectations: linearity!

**Theorem**

For any two random variables $X$ and $Y$, and any constants $\alpha$ and $\beta$:

$E[\alpha X + \beta Y] = \alpha E[X] + \beta E[Y]$ 

Consider rolling two dice. Let $X$ be sum. What is $E[X]$?

- $E[X] = \sum_{e \in \Omega} X(e) \Pr[e]$. 36 term sum!
- $E[X] = \sum_{y \in \mathbb{R}} y \cdot \Pr[X = y]$. What is $\Pr[X = 2]$, $\Pr[X = 3]$, ...?

Instead: $X = X_1 + X_2$. So $E[X] = E[X_1 + X_2] = E[X_1] + E[X_2]$
Linearity of Expectations

Amazing feature of expectations: linearity!

Theorem

For any two random variables $X$ and $Y$, and any constants $\alpha$ and $\beta$:

$$E[\alpha X + \beta Y] = \alpha E[X] + \beta E[Y]$$

Consider rolling two dice. Let $X$ be sum. What is $E[X]$?

- $E[X] = \sum_{e \in \Omega} X(e) \Pr[e]$. 36 term sum!
- $E[X] = \sum_{y \in \mathbb{R}} y \cdot \Pr[X = y]$. What is $\Pr[X = 2]$, $\Pr[X = 3]$, ...?

Instead: $X = X_1 + X_2$. So $E[X] = E[X_1 + X_2] = E[X_1] + E[X_2]$

$$E[X_1] = E[X_2] = \sum_{y=1}^{6} \frac{1}{6}y = \frac{21}{6} = 3.5$$
Linearity of Expectations

Amazing feature of expectations: linearity!

**Theorem**

For any two random variables $X$ and $Y$, and any constants $\alpha$ and $\beta$:

$$E[\alpha X + \beta Y] = \alpha E[X] + \beta E[Y]$$

Consider rolling two dice. Let $X$ be sum. What is $E[X]$?

- $E[X] = \sum_{e \in \Omega} X(e) Pr[e]$. 36 term sum!
- $E[X] = \sum_{y \in \mathbb{R}} y \cdot Pr[X = y]$. What is $Pr[X = 2]$, $Pr[X = 3]$, ... ?

Instead: $X = X_1 + X_2$. So $E[X] = E[X_1 + X_2] = E[X_1] + E[X_2]$

$$E[X_1] = E[X_2] = \sum_{y=1}^{6} \frac{1}{6}y = \frac{21}{6} = 3.5$$

$$\implies E[X] = 3.5 + 3.5 = 7$$
Linearity of Expectations: Proof

**Theorem**

For any two random variables $X$ and $Y$, and any constants $\alpha$ and $\beta$:

$$E[\alpha X + \beta Y] = \alpha E[X] + \beta E[Y]$$

**Proof.**

$$E[\alpha X + \beta Y] = \sum_{e \in \Omega} \Pr[e] (\alpha X(e) + \beta Y(e))$$

Holds no matter how correlated $X$ and $Y$ are!
**Theorem**

For any two random variables $X$ and $Y$, and any constants $\alpha$ and $\beta$:

$$E[\alpha X + \beta Y] = \alpha E[X] + \beta E[Y]$$

**Proof.**

$$E[\alpha X + \beta Y] = \sum_{e \in \Omega} Pr[e] (\alpha X(e) + \beta Y(e))$$

$$= \alpha \sum_{e \in \Omega} Pr[e] X(e) + \beta \sum_{e \in \Omega} Pr[e] X(e)$$

Holds no matter how correlated $X$ and $Y$ are!
**Linearity of Expectations: Proof**

<table>
<thead>
<tr>
<th>Theorem</th>
</tr>
</thead>
</table>
| For any two random variables $X$ and $Y$, and any constants $\alpha$ and $\beta$:  
$E[\alpha X + \beta Y] = \alpha E[X] + \beta E[Y]$ |

<table>
<thead>
<tr>
<th>Proof.</th>
</tr>
</thead>
</table>
| $E[\alpha X + \beta Y] = \sum_{e \in \Omega} \Pr[e] \left( \alpha X(e) + \beta Y(e) \right)$  
$= \alpha \sum_{e \in \Omega} \Pr[e]X(e) + \beta \sum_{e \in \Omega} \Pr[e]X(e)$  
$\text{Holds no matter how correlated } X \text{ and } Y \text{ are!}$ |

<table>
<thead>
<tr>
<th>Proof.</th>
</tr>
</thead>
</table>
| $E[\alpha X + \beta Y] = \sum_{e \in \Omega} \Pr[e] \left( \alpha X(e) + \beta Y(e) \right)$  
$= \alpha \sum_{e \in \Omega} \Pr[e]X(e) + \beta \sum_{e \in \Omega} \Pr[e]X(e)$  
$\text{Holds no matter how correlated } X \text{ and } Y \text{ are!}$ |
Linearity of Expectations: Proof

**Theorem**

For any two random variables $X$ and $Y$, and any constants $\alpha$ and $\beta$:

$$E[\alpha X + \beta Y] = \alpha E[X] + \beta E[Y]$$

**Proof.**

$$E[\alpha X + \beta Y] = \sum_{e \in \Omega} \Pr[e] (\alpha X(e) + \beta Y(e))$$

$$= \alpha \sum_{e \in \Omega} \Pr[e] X(e) + \beta \sum_{e \in \Omega} \Pr[e] X(e)$$

$$= \alpha E[X] + \beta E[Y]$$

Holds no matter how correlated $X$ and $Y$ are!
Randomized Quicksort I

Theorem

The expected running time of randomized quicksort is at most $O(n \log n)$. 
Theorem

The expected running time of randomized quicksort is at most $O(n \log n)$.

Assume for simplicity all elements distinct. Running time $= \Theta(\# \text{ of comparisons})$
Randomized Quicksort I

Theorem

The expected running time of randomized quicksort is at most $O(n \log n)$.

Assume for simplicity all elements distinct. Running time $= \Theta(\# \text{ of comparisons})$

Definitions:

- $X = \# \text{ of comparisons (random variable)}$
- $e_i = i$’th smallest element (for $i \in \{1, \ldots, n\}$)
- $X_{ij}$ random variable for all $i, j \in \{1, \ldots, n\}$ with $i < j$:

$$X_{ij} = \begin{cases} 1 & \text{if algorithm compares } e_i \text{ and } e_j \text{ at any point in time} \\ 0 & \text{otherwise} \end{cases}$$
Randomized Quicksort II

Algorithm never compares the same two elements more than once

$$X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}$$

$$\forall e \in \mathbb{R}, \ X(e) = \sum_{i,j} X_{ij}: (e)$$
Randomized Quicksort II

Algorithm never compares the same two elements more than once

\[ X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij} \]

\[ E[X] = E \left[ \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij} \right] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} E[X_{ij}] \]

So just need to understand \( E[X_{ij}] \)

Simple cases:

1. \( j = i+1 \): \( X_{ij} = 1 \) no matter what, so \( E[X_{ij}] = 1 \)
2. \( i = 1, j = n \): \( e_1 \) and \( e_n \) compared if and only if first pivot chosen is \( e_1 \) or \( e_n \)

So really expectation of \( X_{1n} \)
Randomized Quicksort II

Algorithm never compares the same two elements more than once

\[ X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij} \]

\[
E[X] = E \left[ \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij} \right] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} E[X_{ij}]
\]

So just need to understand \( E[X_{ij}] \)
Algorithm never compares the same two elements more than once

\[ X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij} \]

\[
E[X] = E \left[ \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij} \right] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} E[X_{ij}]
\]

So just need to understand \( E[X_{ij}] \)

Simple cases:
Randomized Quicksort II

Algorithm never compares the same two elements more than once

\[ X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij} \]

\[ E[X] = E\left[ \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij} \right] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} E[X_{ij}] \]

So just need to understand \( E[X_{ij}] \)

Simple cases:

- \( j = i + 1 \):

\[ E[X_{ij}] = 1 \cdot P(X_{ij} = 1) \]
Randomized Quicksort II

Algorithm never compares the same two elements more than once

\[ X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij} \]

\[ E[X] = E \left[ \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij} \right] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} E[X_{ij}] \]

So just need to understand \( E[X_{ij}] \)

Simple cases:

- \( j = i + 1 \): \( X_{ij} = 1 \) no matter what, so \( E[X_{ij}] = 1 \)
Randomized Quicksort II

Algorithm never compares the same two elements more than once

\[ X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij} \]

\[
E[X] = E \left[ \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij} \right] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} E[X_{ij}] 
\]

So just need to understand \( E[X_{ij}] \)

Simple cases:
- \( j = i + 1 \): \( X_{ij} = 1 \) no matter what, so \( E[X_{ij}] = 1 \)
- \( i = 1, j = n \):
Randomized Quicksort II

Algorithm never compares the same two elements more than once

\[ X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij} \]

\[
E[X] = E \left[ \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij} \right] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} E[X_{ij}] 
\]

So just need to understand \( E[X_{ij}] \)

Simple cases:

- \( j = i + 1 \): \( X_{ij} = 1 \) no matter what, so \( E[X_{ij}] = 1 \)
- \( i = 1, j = n \): \( e_1 \) and \( e_n \) compared if and only if first pivot chosen is \( e_1 \) or \( e_n \)
  \[ \implies E[X_{1n}] = \frac{2}{n} \]
$E[X_{ij}]$: General Case ($i < j$)

If $p = e_i$ or $p = e_j$:

If $e_i < p < e_j$:

If $p < e_i$ or $p > e_j$:

Condition on $e_i \leq p \leq e_j$:

Condition on $p \in [e_i, e_j]$:

So $X_{ij}$ not determined until $e_i \leq p \leq e_j$, and when this is determined has $E[X_{ij}] = 2j - i + 1$. 

$\Rightarrow E[X_{ij}] = 2j - i + 1$
**E[X_{ij}]: General Case (i < j)**

If \( p = e_i \) or \( p = e_j \): \( X_{ij} = 1 \)
**E[X_{ij}]: General Case (i < j)**

- If $p = e_i$ or $p = e_j$: $X_{ij} = 1$
- If $e_i < p < e_j$:
\[ E[X_{ij}] : \text{General Case (} i < j \text{)} \]

If \( p = e_i \) or \( p = e_j \): \( X_{ij} = 1 \)

If \( e_i < p < e_j \): \( X_{ij} = 0 \)
$E[X_{ij}]$: General Case ($i < j$)

If $p = e_i$ or $p = e_j$: $X_{ij} = 1$

If $e_i < p < e_j$: $X_{ij} = 0$

If $p < e_i$ or $p > e_j$: 

---

Condition on $e_i \leq p \leq e_j$:

$E[X_{ij}] = 2j - i + 1$

Condition on $p \in [e_i, e_j]$:

still undetermined

So $X_{ij}$ not determined until $e_i \leq p \leq e_j$, and then determined as $E[X_{ij}] = 2j - i + 1$
$E[X_{ij}]$: General Case ($i < j$)

If $p = e_i$ or $p = e_j$: $X_{ij} = 1$

If $e_i < p < e_j$: $X_{ij} = 0$

If $p < e_i$ or $p > e_j$: Both $e_i$, $e_j$ in same recursive call.
$E[X_{ij}]$: General Case ($i < j$)

If $p = e_i$ or $p = e_j$: $X_{ij} = 1$

If $e_i < p < e_j$: $X_{ij} = 0$

If $p < e_i$ or $p > e_j$: ? Both $e_i$, $e_j$ in same recursive call.

- Condition on $e_i \leq p \leq e_j$: 
\[ E[X_{ij}] \]: General Case \((i < j)\)

1. If \(p = e_i \) or \(p = e_j\): \(X_{ij} = 1\)
2. If \(e_i < p < e_j\): \(X_{ij} = 0\)
3. If \(p < e_i \) or \(p > e_j\): ? Both \(e_i\), \(e_j\) in same recursive call.
   - Condition on \(e_i \leq p \leq e_j\): \(E[X_{ij} \mid e_i \leq p \leq e_j] = \frac{2}{j-i+1}\)
\( \mathbb{E}[X_{ij}] \): General Case (\( i < j \))

- If \( p = e_i \) or \( p = e_j \): \( X_{ij} = 1 \)
- If \( e_i < p < e_j \): \( X_{ij} = 0 \)
- If \( p < e_i \) or \( p > e_j \): ? Both \( e_i, e_j \) in same recursive call.
  - Condition on \( e_i \leq p \leq e_j \): \( \mathbb{E}[X_{ij} \mid e_i \leq p \leq e_j] = \frac{2}{j-i+1} \)
  - Condition on \( p \notin [e_i, e_j] \):
**E[X_{ij}]: General Case (i < j)**

If \( p = e_i \) or \( p = e_j \): \( X_{ij} = 1 \)

If \( e_i < p < e_j \): \( X_{ij} = 0 \)

If \( p < e_i \) or \( p > e_j \): ? Both \( e_i, e_j \) in same recursive call.

- Condition on \( e_i \leq p \leq e_j \): \( E[X_{ij} \mid e_i \leq p \leq e_j] = \frac{2}{j-i+1} \)
- Condition on \( p \notin [e_i, e_j] \): still undetermined

\[
\frac{2}{j-i+1} \cdot \mathbb{P}(x \leq \frac{5-\frac{1}{n}}{n}) \leq c_i \leq \sqrt{\frac{5-\frac{1}{n}}{n}}
\]
**E[X_{ij}]: General Case (i < j)**

If \( p = e_i \) or \( p = e_j \): \( X_{ij} = 1 \)

If \( e_i < p < e_j \): \( X_{ij} = 0 \)

If \( p < e_i \) or \( p > e_j \): ? Both \( e_i, e_j \) in same recursive call.

- Condition on \( e_i \leq p \leq e_j \): \( E[X_{ij} \mid e_i \leq p \leq e_j] = \frac{2}{j-i+1} \)
- Condition on \( p \notin [e_i, e_j] \): still undetermined

So \( X_{ij} \) not determined until \( e_i \leq p \leq e_j \), and when it is determined has \( E[X_{ij}] = \frac{2}{j-i+1} \)

\[\implies E[X_{ij}] = \frac{2}{j-i+1}\]
$E[X_{ij}]$: General Case (formally)

Let $Y_k$ be event that the $k$'th pivot is in $[e_i, e_j]$ and all previous pivots not in $[e_i, e_j]$
$E[X_{ij}]$: General Case (formally)

Let $Y_k$ be event that the $k$’th pivot is in $[e_i, e_j]$ and all previous pivots not in $[e_i, e_j]$.

$\implies$ by definition, the $Y_k$ events are disjoint and partition sample space.

$E[X_{ij}] = \sum_{k=1}^{n} E[X_{ij} | Y_k] \Pr[Y_k]$.

Since the $Y_k$ events are disjoint and partition the sample space, we have:

$E[X_{ij}] = \sum_{k=1}^{n} \frac{2j-i+1}{n} = \frac{2j-i+1}{2}$.
**E[X_{ij}]: General Case (formally)**

Let $Y_k$ be event that the $k$'th pivot is in $[e_i, e_j]$ and all previous pivots not in $[e_i, e_j]$.

$\implies$ by definition, the $Y_k$ events are disjoint and partition sample space.

Showed that $E[X_{ij}|Y_k] = \frac{2}{j-i+1}$ for all $k$. 
**E[X_{ij}]: General Case (formally)**

Let $Y_k$ be event that the $k^{th}$ pivot is in $[e_i, e_j]$ and all previous pivots not in $[e_i, e_j]$  
\[ \implies \text{by definition, the $Y_k$ events are disjoint and partition sample space} \]

Showed that $E[X_{ij}|Y_k] = \frac{2}{j-i+1}$ for all $k$.

\[
E[X_{ij}] = \sum_{k=1}^{n} E[X_{ij}|Y_k] \Pr[Y_k] \\
= \frac{2}{j-i+1} \sum_{k=1}^{n} \Pr[Y_k] \\
= \frac{2}{j-i+1} \\
\]

($Y_k$ disjoint and partition $\Omega$)
Expected running time of randomized quicksort:

\[
E[X] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} E[X_{ij}]
\]  
(linearity of expectations)

\[
= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j - i + 1}
\]

\[
= 2 \sum_{i=1}^{n-1} \left( \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n - i + 1} \right)
\]

\[
\leq 2 \sum_{i=1}^{n-1} H_n
\]

\[
\leq 2nH_n
\]

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