Randomized Algorithms

Lecture 11
October 1, 2015
11.1: Randomized Algorithms
11.2: Some Probability
Ω: Sample space
1. $\Omega$: Sample space
2. $\Omega$: Is a set of **elementary event**/atomic event/simple event.
1. \( \Omega \): Sample space

2. \( \Omega \): Is a set of **elementary event**/atomic event/simple event.

3. Every atomic event \( x \in \Omega \) has **Probability** \( \Pr[x] \).
1. $\Omega$: Is a set of **elementary event**/atomic event/simple event.

2. Every atomic event $x \in \Omega$ has **Probability** $\Pr[x]$.

3. $X \equiv f(x)$: Random variable associate a value with each atomic event $x \in \Omega$. 

1. Every atomic event $x \in \Omega$ has \textbf{Probability} $Pr[x]$.

2. $X \equiv f(x)$: Random variable associate a value with each atomic event $x \in \Omega$.

3. $E[X]$: \textbf{Expectation}:
   The average value of the random variable $X \equiv f(x)$.
   $E[X] = \sum_{x \in X} f(x) \cdot Pr[X = x]$. 
1. $X \equiv f(x)$: Random variable associate a value with each atomic event $x \in \Omega$.

2. $E[X]$: Expectation:
The average value of the random variable $X \equiv f(x)$.

$$E[X] = \sum_{x \in X} f(x) \cdot \Pr[X = x].$$

3. An event $A \subseteq \Omega$ is a collection of atomic events.

$$\Pr[A] = \sum_{a \in A} \Pr[a].$$

Complement event: $\overline{A} = \Omega \setminus A$. 
1. \( X \equiv f(x) \): Random variable associate a value with each atomic event \( x \in \Omega \).

2. \( E[X] \): **Expectation**: The average value of the random variable \( X \equiv f(x) \).
   \[
   E[X] = \sum_{x \in X} f(x) \cdot \Pr[X = x].
   \]

3. An event \( A \subseteq \Omega \) is a collection of atomic events.
   \[
   \Pr[A] = \sum_{a \in A} \Pr[a].
   \]
   **Complement event**: \( \overline{A} = \Omega \setminus A \).

4. \( A, B \) two events.
**1.** \( E[X] \): **Expectation:**
The average value of the random variable \( X \equiv f(x) \).
\[
E[X] = \sum_{x \in X} f(x) \times \Pr[X = x].
\]

**2.** An event \( A \subseteq \Omega \) is a collection of atomic events.
\[
\Pr[A] = \sum_{a \in A} \Pr[a].
\]
**Complement event:** \( \overline{A} = \Omega \setminus A \).

**3.** \( A, B \) two events.

**4.** \( A \cap B \): The intersection event.

**5.** \( A \cup B \): The union event.
An event $A \subseteq \Omega$ is a collection of atomic events. 
\[ \Pr[A] = \sum_{a \in A} \Pr[a]. \]
Complement event: $\overline{A} = \Omega \setminus A$. 

2. $A$, $B$ two events. 

3. $A \cap B$: The intersection event. 

4. $A \cup B$: The union event. 

5. $\Pr[A \cup B] = \Pr[A] + \Pr[B] - \Pr[A \cap B]$
$\Pr[A \cup B] \leq \Pr[A] + \Pr[B]$. 

1. $A$, $B$ two events.
2. $A \cap B$: The intersection event.
3. $A \cup B$: The union event.
4. $\Pr[A \cup B] = \Pr[A] + \Pr[B] - \Pr[A \cap B]$  
   $\Pr[A \cup B] \leq \Pr[A] + \Pr[B]$.
5. Tell you that $B$ happened.
1. \( A \cap B \): The intersection event.
2. \( A \cup B \): The union event.
3. \[ \Pr[A \cup B] = \Pr[A] + \Pr[B] - \Pr[A \cap B] \]
   \[ \Pr[A \cup B] \leq \Pr[A] + \Pr[B]. \]
4. Tell you that \( B \) happened.
5. …then what is the probability that \( A \) happened?

**Conditional probability**

\[ \Pr[A \mid B] = \frac{\Pr[A \cap B]}{\Pr[B]} . \]
1. \( A \cup B \): The union event.
2. \[ \Pr[A \cup B] = \Pr[A] + \Pr[B] - \Pr[A \cap B] \]
   \( \Pr[A \cup B] \leq \Pr[A] + \Pr[B] \).
3. Tell you that \( B \) happened.
4. ...then what is the probability that \( A \) happened?
   **Conditional probability**
   \[ \Pr[A \mid B] = \frac{\Pr[A \cap B]}{\Pr[B]} \]
1. \( \Pr[A \cup B] = \Pr[A] + \Pr[B] - \Pr[A \cap B] \)
\( \Pr[A \cup B] \leq \Pr[A] + \Pr[B] \).

2. Tell you that \( B \) happened.

3. ...then what is the probability that \( A \) happened?

**Conditional probability**
\[ \Pr[A \mid B] = \frac{\Pr[A \cap B]}{\Pr[B]} \]
1. Tell you that $B$ happened.
2. ...then what is the probability that $A$ happened?

**Conditional probability**

$$\Pr[A \mid B] = \frac{\Pr[A \cap B]}{\Pr[B]}.$$
**Definition (Informal)**

**Random variable**: a function from probability space to $\mathbb{R}$. Associates value $\forall$ atomic events in probability space.

**Definition**

The conditional probability of $X$ given $Y$ is

$$ \Pr[X = x \mid Y = y] = \frac{\Pr[(X = x) \cap (Y = y)]}{\Pr[Y = y]}.$$

Equivalent to

$$ \Pr[(X = x) \cap (Y = y)] = \Pr[X = x \mid Y = y] \times \Pr[Y = y].$$
Definition (Informal)

**Random variable**: a function from probability space to \( \mathbb{R} \). Associates value \( \forall \) atomic events in probability space.

Definition

The **conditional probability** of \( X \) given \( Y \) is

\[
\Pr \left[ X = x \mid Y = y \right] = \frac{\Pr \left[ (X = x) \cap (Y = y) \right]}{\Pr \left[ Y = y \right]}
\]

Equivalent to

\[
\Pr \left[ (X = x) \cap (Y = y) \right] = \Pr \left[ X = x \mid Y = y \right] \times \Pr \left[ Y = y \right]
\]
Probability - quick review

Definitions

Definition (Informal)

**Random variable**: a function from probability space to \( \mathbb{R} \). Associates value \( \forall \) atomic events in probability space.

Definition

The **conditional probability** of \( X \) given \( Y \) is

\[
\Pr\left[ X = x \mid Y = y \right] = \frac{\Pr\left[ (X = x) \cap (Y = y) \right]}{\Pr\left[ Y = y \right]}
\]

Equivalent to

\[
\Pr\left[ (X = x) \cap (Y = y) \right] = \Pr\left[ X = x \mid Y = y \right] \cdot \Pr\left[ Y = y \right].
\]
The events $X = x$ and $Y = y$ are **independent**, if

$$
\Pr[X = x \cap Y = y] = \Pr[X = x] \cdot \Pr[Y = y].
$$

---

The **expectation** of a random variable $X$ its average value:

$$
E[X] = \sum_{x} x \cdot \Pr[X = x],
$$
Definition

The events \( X = x \) and \( Y = y \) are independent, if

\[
\Pr[ X = x \cap Y = y ] = \Pr[ X = x ] \cdot \Pr[ Y = y ].
\]

\[\equiv \Pr[ X = x \mid Y = y ] = \Pr[ X = x ].\]

Definition

The expectation of a random variable \( X \) is its average value:

\[
E[ X ] = \sum_x x \cdot \Pr[ X = x ],
\]
Lemma (Linearity of expectation.)

∀ random variables $X$ and $Y$: $\mathbb{E}[X + Y] = \mathbb{E}[X] + \mathbb{E}[Y]$.

Proof.

Use definitions, do the math. See notes for details.
Definition

$X, Y$: random variables. The **conditional expectation** of $X$ given $Y$ (i.e., you know $Y = y$):

$$E[X \mid Y] = E[X \mid Y = y] = \sum_x x \cdot \Pr(X = x \mid Y = y).$$

$E[X]$ is a number.

$f(y) = E[X \mid Y = y]$ is a function.
Lemma

∀ \( X, Y \) (not necessarily independent): \( E[X] = E[E[X | Y]] \).

\[
E[E[X | Y]] = E_y [E[X | Y = y]]
\]

Proof.

Use definitions, and do the math. See class notes.
Conditional Expectation

Lemma

∀ X, Y (not necessarily independent): \( E[X] = E\left[ E\left[ X \mid Y \right] \right] \).

\[ E\left[ E\left[ X \mid Y \right] \right] = E_y \left[ E\left[ X \mid Y = y \right] \right] \]

Proof.

Use definitions, and do the math. See class notes.
Lemma

∀ X, Y (not necessarily independent): \( \mathbb{E}[X] = \mathbb{E}[\mathbb{E}[X | Y]] \).

\[ \mathbb{E}[\mathbb{E}[X | Y]] = \mathbb{E}_y \mathbb{E}[X | Y = y] \]

Proof.

Use definitions, and do the math. See class notes.
11.3: Sorting Nuts and Bolts
Problem (Sorting Nuts and Bolts)

1. **Input:** Set $n$ nuts + $n$ bolts.
2. Every nut have a matching bolt.
3. All different sizes.
4. **Task:** Match nuts to bolts. (In sorted order).
5. **Restriction:** You can only compare a nut to a bolt.
6. **Q:** How to match the $n$ nuts to the $n$ bolts quickly?
Problem (Sorting Nuts and Bolts)

1. Input: Set $n$ nuts + $n$ bolts.
2. Every nut have a matching bolt.
3. All different sizes.
4. Task: Match nuts to bolts. (In sorted order).
5. Restriction: You can only compare a nut to a bolt.
6. Q: How to match the $n$ nuts to the $n$ bolts quickly?
Problem \textbf{(Sorting Nuts and Bolts)}

1. \textit{Input:} Set $n$ nuts + $n$ bolts.
2. Every nut have a matching bolt.
3. All different sizes.
4. \textit{Task:} Match nuts to bolts. (In sorted order).
5. Restriction: You can only compare a nut to a bolt.
6. Q: How to match the $n$ nuts to the $n$ bolts quickly?
Sorting Nuts & Bolts

Problem (**Sorting Nuts and Bolts**)

1. **Input:** Set $n$ nuts + $n$ bolts.
2. Every nut have a matching bolt.
3. All different sizes.
4. **Task:** Match nuts to bolts. (In sorted order).
5. **Restriction:** You can only compare a nut to a bolt.
6. **Q:** How to match the $n$ nuts to the $n$ bolts quickly?
Problem (Sorting Nuts and Bolts)

1. **Input:** Set \( n \) nuts + \( n \) bolts.
2. Every nut have a matching bolt.
3. All different sizes.
4. **Task:** Match nuts to bolts. (In sorted order).
5. **Restriction:** You can only compare a nut to a bolt.
6. **Q:** How to match the \( n \) nuts to the \( n \) bolts quickly?
Sorting nuts & bolts...

Algorithm

1. Naive algorithm...
2. ...better algorithm?
Sorting nuts & bolts...

Algorithm

1. Naive algorithm...
2. ...better algorithm?
Sorting nuts & bolts...

\textbf{MatchNutsAndBolts}(N: nuts, B: bolts)

Pick a random nut \(n_{\text{pivot}}\) from \(N\)
Find its matching bolt \(b_{\text{pivot}}\) in \(B\)
\(B_L \leftarrow \) All bolts in \(B\) smaller than \(n_{\text{pivot}}\)
\(N_L \leftarrow \) All nuts in \(N\) smaller than \(b_{\text{pivot}}\)
\(B_R \leftarrow \) All bolts in \(B\) larger than \(n_{\text{pivot}}\)
\(N_R \leftarrow \) All nuts in \(N\) larger than \(b_{\text{pivot}}\)
\text{MatchNutsAndBolts}(N_R,B_R)
\text{MatchNutsAndBolts}(N_L,B_L)

QuickSort style...
Sorting nuts & bolts...

**MatchNutsAndBolts**($N$: nuts, $B$: bolts)

Pick a random nut $n_{\text{pivot}}$ from $N$
Find its matching bolt $b_{\text{pivot}}$ in $B$

$B_L \leftarrow$ All bolts in $B$ smaller than $n_{\text{pivot}}$
$N_L \leftarrow$ All nuts in $N$ smaller than $b_{\text{pivot}}$
$B_R \leftarrow$ All bolts in $B$ larger than $n_{\text{pivot}}$
$N_R \leftarrow$ All nuts in $N$ larger than $b_{\text{pivot}}$

MatchNutsAndBolts($N_R, B_R$)
MatchNutsAndBolts($N_L, B_L$)

QuickSort style...
11.3.1: Running time analysis
What is running time for randomized algorithms?

Definitions

**Definition**

$RT(U)$: random variable – **running time** of the algorithm on input $U$.

**Definition**

Expected running time $E[RT(U)]$ for input $U$.

**Definition**

Expected **running-time** of algorithm for input size $n$:

$$T(n) = \max_{U \text{ is an input of size } n} E[RT(U)].$$
What is running time for randomized algorithms?

Definitions

**Definition**

$RT(U)$: random variable – **running time** of the algorithm on input $U$.

**Definition**

Expected running time $E[RT(U)]$ for input $U$.

**Definition**

expected running-time of algorithm for input size $n$:

$$T(n) = \max_{U \text{ is an input of size } n} E[RT(U)].$$
What is running time for randomized algorithms?

Definitions

**Definition**

\( \mathcal{RT}(U) \): random variable – **running time** of the algorithm on input \( U \).

**Definition**

Expected running time \( \mathbb{E}[\mathcal{RT}(U)] \) for input \( U \).

**Definition**

expected **running-time** of algorithm for input size \( n \):

\[
T(n) = \max_{U \text{ is an input of size } n} \mathbb{E}[\mathcal{RT}(U)].
\]
What is running time for randomized algorithms?

More definitions

**Definition**

$\text{rank}(x)$: rank of element $x \in S =$ number of elements in $S$ smaller or equal to $x$. 
Theorem

Expected running time \textbf{MatchNutsAndBolts (QuickSort)} is
\[ T(n) = O(n \log n). \text{ Worst case is } O(n^2). \]

Proof.

\[ \Pr[\text{rank}(n_{\text{pivot}}) = k] = \frac{1}{n}. \text{ Thus,} \]

\[ T(n) = \mathbb{E}_{k=\text{rank}(n_{\text{pivot}})} \left[ O(n) + T(k - 1) + T(n - k) \right] \]
Expected running time MatchNutsAndBolts (QuickSort) is $T(n) = \Theta(n \log n)$. Worst case is $O(n^2)$.

Proof.

$$\Pr[\text{rank}(n_{pivot}) = k] = \frac{1}{n}. \ \text{Thus,}$$

$$T(n) = \mathbb{E}_{k=\text{rank}(n_{pivot})} \left[ O(n) + T(k - 1) + T(n - k) \right]$$

$$= O(n) + \mathbb{E}_k[T(k - 1) + T(n - k)]$$
**Theorem**

*Expected running time* **MatchNutsAndBolts** (*QuickSort*) *is*

\[ T(n) = O(n \log n). \text{ Worst case is } O(n^2). \]

**Proof.**

\[ \Pr[\text{rank}(n_{pivot}) = k] = \frac{1}{n}. \text{ Thus,} \]

\[
T(n) = O(n) + \mathbb{E}_{k}
\left[
T(k - 1) + T(n - k)
\right]
\]
Theorem

Expected running time **MatchNutsAndBolts (QuickSort)** is

\[ T(n) = O(n \log n). \text{ Worst case is } O(n^2). \]

Proof.

\[ \Pr[\text{rank}(n_{pivot}) = k] = \frac{1}{n}. \text{ Thus,} \]

\[ T(n) = O(n) + \sum_{k=1}^{n} \Pr[\text{Rank}(\text{Pivot}) = k] \]

\[ = O(n) + \sum_{k=1}^{n} \Pr[\text{Rank}(\text{Pivot}) = k] \]

\[ \times (T(k - 1) + T(n - k)) \]
Theorem

Expected running time \textbf{MatchNutsAndBolts (QuickSort)} is \(T(n) = O(n \log n)\). Worst case is \(O(n^2)\).

Proof.

\( \Pr[\text{rank}(n_{\text{pivot}}) = k] = \frac{1}{n} \). Thus,

\[
T(n) = O(n) + \sum_{k=1}^{n} \Pr[\text{Rank(Pivot)} = k] \times (T(k - 1) + T(n - k))
\]
Expected running time \textbf{MatchNutsAndBolts (QuickSort)} is 
\[ T(n) = O(n \log n). \] Worst case is \( O(n^2). \)

\textbf{Proof.}

\[ \Pr[\text{rank}(n_{pivot}) = k] = \frac{1}{n}. \] Thus,
\[ T(n) = O(n) + \sum_{k=1}^{n} \Pr[\text{Rank(Pivot)} = k] \]
\[ \times (T(k - 1) + T(n - k)) \]
\[ = O(n) + \sum_{k=1}^{n} \frac{1}{n} \cdot (T(k - 1) + T(n - k)), \]
Nuts and bolts running time

**Theorem**

*Expected running time* \( \text{MatchNutsAndBolts (QuickSort)} \) is

\[ T(n) = O(n \log n). \text{ Worst case is } O(n^2). \]

**Proof.**

\[ \Pr[\text{rank}(n_{pivot}) = k] = \frac{1}{n}. \text{ Thus,} \]

\[ T(n) = O(n) + \sum_{k=1}^{n} \frac{1}{n} \cdot (T(k - 1) + T(n - k)), \]
Nuts and bolts running time

**Theorem**

*Expected running time* MatchNutsAndBolts (*QuickSort*) *is*

\[ T(n) = O(n \log n). \text{ Worst case is } O(n^2). \]

**Proof.**

\[ \Pr[\text{rank}(n_{pivot}) = k] = \frac{1}{n}. \text{ Thus,} \]

\[ T(n) = O(n) + \sum_{k=1}^{n} \frac{1}{n} \cdot (T(k - 1) + T(n - k)), \]

Solution is \( T(n) = O(n \log n). \)
11.3.1.1: Alternative incorrect solution
Alternative intuitive analysis...

Which is not formally correct

1. **MatchNutsAndBolts** is **lucky** if \( \frac{n}{4} \leq \text{rank}(n_{pivot}) \leq \frac{3}{4}n \).

2. \( \Pr[\text{“lucky”}] = 1/2 \).

3. \( T(n) \leq O(n) + \Pr[\text{“lucky”}] \times (T(n/4) + T(3n/4)) + \Pr[\text{“unlucky”}] \times T(n) \).

4. \( T(n) = O(n) + \frac{1}{2} \times (T(\frac{n}{4}) + T(\frac{3}{4}n)) + \frac{1}{2}T(n) \).

5. Rewriting: \( T(n) = O(n) + T(n/4) + T((3/4)n) \).

6. ... solution is \( O(n \log n) \).
Alternative intuitive analysis...
Which is not formally correct

1. **MatchNutsAndBolts** is lucky if \( \frac{n}{4} \leq \text{rank}(n_{\text{pivot}}) \leq \frac{3}{4}n \).
2. \( \Pr[\text{“lucky”}] = 1/2 \).
3. \( T(n) \leq O(n) + \Pr[\text{“lucky”}] \times (T(n/4) + T(3n/4)) + \Pr[\text{“unlucky”}] \times T(n) \).
4. \( T(n) = O(n) + \frac{1}{2} \times (T(n/4) + T(3n/4)) + \frac{1}{2}T(n) \).
5. Rewriting: \( T(n) = O(n) + T(n/4) + T((3/4)n) \).
6. ... solution is \( O(n \log n) \).
Alternative intuitive analysis...

Which is not formally correct

1. **MatchNutsAndBolts** is **lucky** if \( \frac{n}{4} \leq \text{rank}(n_{pivot}) \leq \frac{3}{4}n \).

2. \( \Pr["lucky"] = 1/2. \)

3. \( T(n) \leq O(n) + \Pr["lucky"] \times (T(n/4) + T(3n/4)) + \Pr["unlucky"] \times T(n). \)

4. \( T(n) = O(n) + \frac{1}{2} \times (T(\frac{n}{4}) + T(\frac{3}{4}n)) + \frac{1}{2}T(n). \)

5. **Rewriting**: \( T(n) = O(n) + T(n/4) + T((3/4)n). \)

6. **... solution is** \( O(n \log n). \)
Alternative intuitive analysis...

Which is not formally correct

1. \textbf{MatchNutsAndBolts} is \textbf{lucky} if $\frac{n}{4} \leq \text{rank}(n_{\text{pivot}}) \leq \frac{3}{4}n$.

2. $\Pr[\text{“lucky”}] = 1/2$.

3. $T(n) \leq O(n) + \Pr[\text{“lucky”}] \times (T(n/4) + T(3n/4)) + \Pr[\text{“unlucky”}] \times T(n)$.

4. $T(n) = O(n) + \frac{1}{2} \times (T(\frac{n}{4}) + T(\frac{3}{4}n)) + \frac{1}{2}T(n)$.

5. Rewriting: $T(n) = O(n) + T(n/4) + T((3/4)n)$.

6. ... solution is $O(n \log n)$. 
Alternative intuitive analysis...
Which is not formally correct

1. **MatchNutsAndBolts** is **lucky** if \( \frac{n}{4} \leq \text{rank}(n_{\text{pivot}}) \leq \frac{3}{4}n \).
2. \( \Pr[\text{“lucky”}] = \frac{1}{2} \).
3. \( T(n) \leq O(n) + \Pr[\text{“lucky”}] \ast (T(n/4) + T(3n/4)) + \Pr[\text{“unlucky”}] \ast T(n) \).
4. \( T(n) = O(n) + \frac{1}{2} \ast (T(\frac{n}{4}) + T(\frac{3}{4}n)) + \frac{1}{2}T(n) \).
5. **Rewriting**: \( T(n) = O(n) + T(n/4) + T((3/4)n) \).

6. ... solution is \( O(n \log n) \).
Alternative intuitive analysis...

Which is not formally correct

1. **MatchNutsAndBolts** is **lucky** if \( \frac{n}{4} \leq \text{rank}(n_{pivot}) \leq \frac{3}{4}n \).

2. \( \Pr[\text{“lucky”}] = 1/2. \)

3. \( T(n) \leq O(n) + \Pr[\text{“lucky”}] \times (T(n/4) + T(3n/4)) + \Pr[\text{“unlucky”}] \times T(n). \)

4. \( T(n) = O(n) + \frac{1}{2} \times (T(\frac{n}{4}) + T(\frac{3}{4}n)) + \frac{1}{2}T(n). \)

5. Rewriting: \( T(n) = O(n) + T(n/4) + T((3/4)n). \)

6. ... solution is \( O(n \log n). \)
11.3.2: What are randomized algorithms?
Worst case vs. average case

Expected running time of a randomized algorithm is

\[ T(n) = \max_{U \text{ is an input of size } n} \mathbb{E}[\mathcal{R} \mathcal{T}(U)], \]

Worst case running time of deterministic algorithm:

\[ T(n) = \max_{U \text{ is an input of size } n} \mathcal{R} \mathcal{T}(U), \]
High Probability running time...

**Definition**

Running time $\text{Alg}$ is $O(f(n))$ with **high probability** if

$$\Pr\left[ \mathcal{RT}(\text{Alg}(n)) \geq c \cdot f(n) \right] = o(1).$$

$$\Rightarrow \Pr\left[ \mathcal{RT}(\text{Alg}) > c \cdot f(n) \right] \to 0 \text{ as } n \to \infty.$$

Usually use weaker def:

$$\Pr\left[ \mathcal{RT}(\text{Alg}(n)) \geq c \cdot f(n) \right] \leq \frac{1}{n^d},$$

Technical reasons... also assume that $\mathbb{E}[\mathcal{RT}(\text{Alg}(n))] = O(f(n))$. 

---

Sariel (UIUC) New CS473 Fall 2015 22 / 30
Definition

Running time $\text{Alg}$ is $O(f(n))$ with high probability if

$$\Pr\left[\mathcal{RT}(\text{Alg}(n)) \geq c \cdot f(n)\right] = o(1).$$

$$\implies \Pr\left[\mathcal{RT}(\text{Alg}) > c \cdot f(n)\right] \to 0 \text{ as } n \to \infty.$$

Usually use weaker def:

$$\Pr\left[\mathcal{RT}(\text{Alg}(n)) \geq c \cdot f(n)\right] \leq \frac{1}{n^d},$$

Technical reasons... also assume that $\mathbb{E}[\mathcal{RT}(\text{Alg}(n))] = O(f(n)).$
Definition

Running time $\text{Alg}$ is $O(f(n))$ with high probability if

$$\Pr \left[ \mathcal{RT}(\text{Alg}(n)) \geq c \cdot f(n) \right] = o(1).$$

$$\Rightarrow \Pr \left[ \mathcal{RT}(\text{Alg}) > c \ast f(n) \right] \to 0 \text{ as } n \to \infty.$$ 

Usually use weaker def:

$$\Pr \left[ \mathcal{RT}(\text{Alg}(n)) \geq c \cdot f(n) \right] \leq \frac{1}{n^d},$$

Technical reasons... also assume that $\mathbb{E}[\mathcal{RT}(\text{Alg}(n))] = O(f(n))$. 
High Probability running time...

**Definition**

Running time \( \text{Alg} \) is \( O(f(n)) \) with **high probability** if

\[
\Pr \left[ \mathcal{R}_T(\text{Alg}(n)) \geq c \cdot f(n) \right] = o(1).
\]

\[
\Rightarrow \quad \Pr \left[ \mathcal{R}_T(\text{Alg}) > c \cdot f(n) \right] \to 0 \text{ as } n \to \infty.
\]

Usually use weaker def:

\[
\Pr \left[ \mathcal{R}_T(\text{Alg}(n)) \geq c \cdot f(n) \right] \leq \frac{1}{n^d},
\]

Technical reasons... also assume that \( \mathbb{E}[\mathcal{R}_T(\text{Alg}(n))] = O(f(n)) \).
11.4: Slick analysis of QuickSort
A Slick Analysis of QuickSort

Let \( Q(A) \) be number of comparisons done on input array \( A \):

1. For \( 1 \leq i < j < n \) let \( R_{ij} \) be the event that rank \( i \) element is compared with rank \( j \) element.

2. \( X_{ij} \): indicator random variable for \( R_{ij} \).
   \( X_{ij} = 1 \iff \) rank \( i \) element compared with rank \( j \) element, otherwise \( 0 \).

\[
Q(A) = \sum_{1 \leq i < j \leq n} X_{ij}
\]

and hence by linearity of expectation,

\[
E[Q(A)] = \sum_{1 \leq i < j \leq n} E[X_{ij}] = \sum_{1 \leq i < j \leq n} \Pr[R_{ij}].
\]
A Slick Analysis of **QuickSort**

Let $Q(A)$ be number of comparisons done on input array $A$:

1. For $1 \leq i < j < n$ let $R_{ij}$ be the event that rank $i$ element is compared with rank $j$ element.

2. $X_{ij}$: **indicator random variable** for $R_{ij}$. $X_{ij} = 1 \iff$ rank $i$ element compared with rank $j$ element, otherwise 0.

$$Q(A) = \sum_{1 \leq i < j \leq n} X_{ij}$$

and hence by linearity of expectation,

$$E\left[ Q(A) \right] = \sum_{1 \leq i < j \leq n} E\left[ X_{ij} \right] = \sum_{1 \leq i < j \leq n} Pr\left[ R_{ij} \right].$$
A Slick Analysis of QuickSort

\( R_{ij} = \text{rank } i \text{ element is compared with rank } j \text{ element.} \)

**Question:** What is \( \Pr[R_{ij}] \)?

7 5 9 1 3 4 8 6
A Slick Analysis of QuickSort

\( R_{ij} = \text{rank } i \text{ element is compared with rank } j \text{ element.} \)

**Question:** What is \( \Pr[R_{ij}] \)?

With ranks: 6 4 8 1 2 3 7 5
A Slick Analysis of **QuickSort**

\[ R_{ij} = \text{rank } i \text{ element is compared with rank } j \text{ element.} \]

**Question:** What is \( \Pr[R_{ij}] \)?

<table>
<thead>
<tr>
<th>7</th>
<th>5</th>
<th>9</th>
<th>1</th>
<th>3</th>
<th>4</th>
<th>8</th>
<th>6</th>
</tr>
</thead>
</table>

With ranks: 6 4 8 1 2 3 7 5

As such, probability of comparing 5 to 8 is \( \Pr[R_{4,7}] \).
$R_{ij} = \text{rank } i \text{ element is compared with rank } j \text{ element.}$

**Question:** What is $\Pr[R_{ij}]$?

With ranks: 6 4 8 1 2 3 7 5

If pivot too small (say 3 [rank 2]). Partition and call recursively:

Decision if to compare 5 to 8 is moved to subproblem.
A Slick Analysis of **QuickSort**

$R_{ij} = \text{rank } i \text{ element is compared with rank } j \text{ element.}$

**Question:** What is $Pr[R_{ij}]$?

With ranks: 6 4 8 1 2 3 7 5

1. If pivot too small (say 3 [rank 2]). Partition and call recursively:

   \[
   \begin{array}{cccccccc}
   7 & 5 & 9 & 1 & 3 & 4 & 8 & 6 \\
   \end{array}
   \Rightarrow
   \begin{array}{cccccccc}
   1 & 3 & 7 & 5 & 9 & 4 & 8 & 6 \\
   \end{array}
   \]

   Decision if to compare 5 to 8 is moved to subproblem.

2. If pivot too large (say 9 [rank 8]):

   \[
   \begin{array}{cccccccc}
   7 & 5 & 9 & 1 & 3 & 4 & 8 & 6 \\
   \end{array}
   \Rightarrow
   \begin{array}{ccccccccc}
   7 & 5 & 1 & 3 & 4 & 8 & 6 & 9 \\
   \end{array}
   \]

   Decision if to compare 5 to 8 moved to subproblem.
A Slick Analysis of QuickSort

**Question:** What is $Pr[R_{i,j}]$?

As such, probability of comparing 5 to 8 is $Pr[R_{4,7}]$.

If pivot is 5 (rank 4). Bingo!

1 3 4 5 7 9 8 6
A Slick Analysis of QuickSort

Question: What is $\Pr[R_{i,j}]$?

As such, probability of comparing 5 to 8 is $\Pr[R_{4,7}]$.

1. If pivot is 5 (rank 4). Bingo!

2. If pivot is 8 (rank 7). Bingo!
A Slick Analysis of **QuickSort**

**Question:** What is $\Pr[R_{i,j}]$?

1. **If pivot is 5** (rank 4). Bingo!

   ![Diagram 1](image1.png)

   As such, probability of comparing 5 to 8 is $\Pr[R_{4,7}]$.

2. **If pivot is 8** (rank 7). Bingo!

   ![Diagram 2](image2.png)

3. **If pivot in between the two numbers** (say 6 [rank 5]):

   ![Diagram 3](image3.png)

   5 and 8 will never be compared to each other.
A Slick Analysis of **QuickSort**

**Question:** What is $\Pr[R_{i,j}]$?

**Conclusion:**

$R_{i,j}$ happens if and only if:

- $i$th or $j$th ranked element is the first pivot out of $i$th to $j$th ranked elements.

**How to analyze this?**

Thinking acrobatics!

1. Assign every element in the array a random priority (say in $[0, 1]$).
2. Choose pivot to be the element with lowest priority in subproblem.
3. Equivalent to picking pivot uniformly at random (as **QuickSort** do).
A Slick Analysis of **QuickSort**

**Question:** What is \( \Pr[R_{i,j}] \)?

**How to analyze this?**

Thinking acrobatics!

1. Assign every element in the array a random priority (say in \([0, 1]\)).
2. Choose pivot to be the element with lowest priority in subproblem.

\[ R_{i,j} \text{ happens if either } i \text{ or } j \text{ have lowest priority out of elements rank } i \text{ to } j, \]

There are \( k = j - i + 1 \) relevant elements.

\[
\Pr[R_{i,j}] = \frac{2}{k} = \frac{2}{j - i + 1}.
\]
A Slick Analysis of **QuickSort**

**Question:** What is $Pr[R_{i,j}]$?

**How to analyze this?**

Thinking acrobatics!

1. Assign every element in the array a random priority (say in $[0, 1]$).
2. Choose pivot to be the element with lowest priority in subproblem.

$\implies R_{i,j}$ happens if either $i$ or $j$ have lowest priority out of elements rank $i$ to $j$,

There are $k = j - i + 1$ relevant elements.

$$Pr[R_{i,j}] = \frac{2}{k} = \frac{2}{j - i + 1}.$$
Question: What is $\Pr[R_{ij}]$?

Lemma

$$\Pr[R_{ij}] = \frac{2}{j-i+1}.$$  

Proof.

Let $a_1, \ldots, a_i, \ldots, a_j, \ldots, a_n$ be elements of $A$ in sorted order.
Let $S = \{a_i, a_{i+1}, \ldots, a_j\}$

Observation: If pivot is chosen outside $S$ then all of $S$ either in left array or right array.

Observation: $a_i$ and $a_j$ separated when a pivot is chosen from $S$ for the first time. Once separated no comparison.

Observation: $a_i$ is compared with $a_j$ if and only if either $a_i$ or $a_j$ is chosen as a pivot from $S$ at separation...
A Slick Analysis of \textbf{QuickSort}

\textbf{Question:} What is $\Pr[R_{ij}]$?

\textbf{Lemma}

$$\Pr[R_{ij}] = \frac{2}{j-i+1}.$$ 

\textbf{Proof.}

Let $a_1, \ldots, a_i, \ldots, a_j, \ldots, a_n$ be elements of $A$ in sorted order. Let $S = \{a_i, a_{i+1}, \ldots, a_j\}$

\textbf{Observation:} If pivot is chosen outside $S$ then all of $S$ either in left array or right array.

\textbf{Observation:} $a_i$ and $a_j$ separated when a pivot is chosen from $S$ for the first time. Once separated no comparison.

\textbf{Observation:} $a_i$ is compared with $a_j$ if and only if either $a_i$ or $a_j$ is chosen as a pivot from $S$ at separation...
A Slick Analysis of QuickSort

**Question:** What is $\Pr[R_{ij}]$?

**Lemma**

$$\Pr[R_{ij}] = \frac{2}{j-i+1}.$$  

**Proof.**  
Let $a_1, \ldots, a_i, \ldots, a_j, \ldots, a_n$ be elements of $A$ in sorted order. Let $S = \{a_i, a_{i+1}, \ldots, a_j\}$

**Observation:** If pivot is chosen outside $S$ then all of $S$ either in left array or right array.

**Observation:** $a_i$ and $a_j$ separated when a pivot is chosen from $S$ for the first time. Once separated no comparison.

**Observation:** $a_i$ is compared with $a_j$ if and only if either $a_i$ or $a_j$ is chosen as a pivot from $S$ at separation...
Lemma

\[ \Pr[R_{ij}] = \frac{2}{j-i+1}. \]

Proof.

Let \( a_1, \ldots, a_i, \ldots, a_j, \ldots, a_n \) be sort of \( A \). Let \( S = \{a_i, a_{i+1}, \ldots, a_j\} \)

Observation: \( a_i \) is compared with \( a_j \) if and only if either \( a_i \) or \( a_j \) is chosen as a pivot from \( S \) at separation.

Observation: Given that pivot is chosen from \( S \) the probability that it is \( a_i \) or \( a_j \) is exactly \( 2/|S| = 2/(j-i+1) \) since the pivot is chosen uniformly at random from the array.
A Slick Analysis of QuickSort

Continued...

\[
\mathbb{E}[Q(A)] = \sum_{1 \leq i < j \leq n} \mathbb{E}[X_{ij}] = \sum_{1 \leq i < j \leq n} \Pr[R_{ij}].
\]

**Lemma**

\[
\Pr[R_{ij}] = \frac{2}{j-i+1}.
\]

\[
\mathbb{E}[Q(A)] = \sum_{1 \leq i < j \leq n} \frac{2}{j-i+1}
\]
Lemma

\[ \Pr[R_{ij}] = \frac{2}{j-i+1}. \]

\[
\mathbb{E}[Q(A)] = \sum_{1 \leq i < j \leq n} \Pr[R_{ij}] = \sum_{1 \leq i < j \leq n} \frac{2}{j-i+1}
\]
A Slick Analysis of QuickSort

Continued...

Lemma

\[ \Pr[R_{ij}] = \frac{2}{j-i+1}. \]

\[ \mathbb{E}[Q(A)] = \sum_{1 \leq i < j \leq n} \frac{2}{j-i+1} \]
A Slick Analysis of QuickSort

Continued...

Lemma

\[ \Pr[R_{ij}] = \frac{2}{j-i+1}. \]

\[
\mathbf{E} \left[ Q(A) \right] = \sum_{1 \leq i < j \leq n} \frac{2}{j-i+1}
\]

\[ = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1} \]
A Slick Analysis of QuickSort

Lemma

\[ \Pr[R_{ij}] = \frac{2}{j-i+1}. \]

\[ \mathbb{E}[Q(A)] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1} \]
Lemma

\[ \Pr[R_{ij}] = \frac{2}{j-i+1}. \]

\[ E[Q(A)] = 2 \sum_{i=1}^{n-1} \sum_{i<j}^{n} \frac{1}{j-i+1} \]
A Slick Analysis of QuickSort

Continued...

Lemma

\[ \Pr[R_{ij}] = \frac{2}{j-i+1}. \]

\[ E[Q(A)] = 2 \sum_{i=1}^{n-1} \sum_{i<j}^{n} \frac{1}{j-i+1} \]
A Slick Analysis of QuickSort

Continued...

Lemma

\[ \Pr[R_{ij}] = \frac{2}{j - i + 1}. \]

\[
E[Q(A)] = 2 \sum_{i=1}^{n-1} \sum_{i<j} \frac{1}{j - i + 1} \leq 2 \sum_{i=1}^{n-1} \sum_{\Delta=2}^{n-i+1} \frac{1}{\Delta}
\]
A Slick Analysis of **QuickSort**

Continued...

**Lemma**

\[
\Pr[R_{ij}] = \frac{2}{j - i + 1}.
\]

\[
\mathbb{E}[Q(A)] = 2 \sum_{i=1}^{n-1} \sum_{j<i} \frac{1}{j - i + 1} \leq 2 \sum_{i=1}^{n-1} \sum_{\Delta=2}^{n-i+1} \frac{1}{\Delta}
\]

\[
\leq 2 \sum_{i=1}^{n-1} (H_{n-i+1} - 1) \leq 2 \sum_{1 \leq i < n} H_n
\]
A Slick Analysis of QuickSort

Continued...

Lemma

\[ \Pr[R_{ij}] = \frac{2}{j-i+1}. \]

\[ E[Q(A)] = 2 \sum_{i=1}^{n-1} \sum_{i<j}^{n} \frac{1}{j-i+1} \leq 2 \sum_{i=1}^{n-1} \sum_{\Delta=2}^{n-i+1} \frac{1}{\Delta} \]

\[ \leq 2 \sum_{i=1}^{n-1} (H_{n-i+1} - 1) \leq 2 \sum_{1 \leq i < n} H_n \]

\[ \leq 2nH_n = O(n \log n) \]
11.5: Quick Select
11.6: Randomized Selection
Randomized Quick Selection

Input  Unsorted array $A$ of $n$ integers, an integer $j$.
Goal   Find the $j$th smallest number in $A$ (rank $j$ number)

Randomized Quick Selection
1. Pick a pivot element *uniformly at random* from the array.
2. Split array into 3 subarrays: those smaller than pivot, those larger than pivot, and the pivot itself.
3. Return pivot if rank of pivot is $j$.
4. Otherwise recurse on one of the arrays depending on $j$ and their sizes.
Algorithm for Randomized Selection

Assume for simplicity that \( A \) has distinct elements.

\textbf{QuickSelect}(\( A, j \)):
Pick pivot \( x \) uniformly at random from \( A \)
Partition \( A \) into \( A_{\text{less}} \), \( x \), and \( A_{\text{greater}} \) using \( x \) as pivot
if \( |A_{\text{less}}| = j - 1 \) then
return \( x \)
if \( |A_{\text{less}}| \geq j \) then
return \textbf{QuickSelect}(\( A_{\text{less}}, j \))
else
return \textbf{QuickSelect}(\( A_{\text{greater}}, j - |A_{\text{less}}| - 1 \))
QuickSelect analysis

1. $S_1, S_2, \ldots, S_k$ be the subproblems considered by the algorithm. Here $|S_1| = n$.

2. $S_i$ would be **successful** if $|S_i| \leq (3/4) |S_{i-1}|$

3. $Y_1 = \text{number of recursive calls till first successful iteration.}$ Clearly, total work till this happens is $O(Y_1 n)$.

4. $n_i = \text{size of the subproblem immediately after the } (i - 1)\text{th successful iteration.}$

5. $Y_i = \text{number of recursive calls after the } (i - 1)\text{th successful call, till the } i\text{th successful iteration.}$

6. Running time is $O(\sum_i n_i Y_i)$.
QuickSelect analysis

Example

\( S_i = \) subarray used in \( i \)th recursive call

\(|S_i| = \) size of this subarray

Red indicates successful iteration.

<table>
<thead>
<tr>
<th>Inst’</th>
<th>( S_1 )</th>
<th>( S_2 )</th>
<th>( S_3 )</th>
<th>( S_4 )</th>
<th>( S_5 )</th>
<th>( S_6 )</th>
<th>( S_7 )</th>
<th>( S_8 )</th>
<th>( S_9 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
<td>S_i</td>
<td>)</td>
<td>100</td>
<td>70</td>
<td>60</td>
<td>50</td>
<td>40</td>
<td>30</td>
<td>25</td>
</tr>
<tr>
<td>Succ’</td>
<td>( Y_1 = 2 )</td>
<td>( Y_2 = 4 )</td>
<td>( Y_3 = 2 )</td>
<td>( Y_4 = 1 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( n_i = )</td>
<td>( n_1 = 100 )</td>
<td>( n_2 = 60 )</td>
<td>( n_3 = 25 )</td>
<td>( n_4 = 2 )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. All the subproblems after \((i - 1)\)th successful iteration till \(i\)th successful iteration have size \( \leq n_i \).

2. Total work: \( O(\sum n_i Y_i) \).
QuickSelect analysis

Example

$S_i =$ subarray used in $i$th recursive call

$|S_i| =$ size of this subarray

Red indicates successful iteration.

<table>
<thead>
<tr>
<th>Inst’</th>
<th>$S_1$</th>
<th>$S_2$</th>
<th>$S_3$</th>
<th>$S_4$</th>
<th>$S_5$</th>
<th>$S_6$</th>
<th>$S_7$</th>
<th>$S_8$</th>
<th>$S_9$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$</td>
<td>S_i</td>
<td>$</td>
<td>100</td>
<td>70</td>
<td>60</td>
<td>50</td>
<td>40</td>
<td>30</td>
<td>25</td>
</tr>
<tr>
<td>Succ’</td>
<td>$Y_1 = 2$</td>
<td></td>
<td>$Y_2 = 4$</td>
<td></td>
<td>$Y_3 = 2$</td>
<td></td>
<td>$Y_4 = 1$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$n_i =$</td>
<td>$n_1 = 100$</td>
<td></td>
<td>$n_2 = 60$</td>
<td></td>
<td>$n_3 = 25$</td>
<td></td>
<td>$n_4 = 2$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. All the subproblems after $(i - 1)$th successful iteration till $i$th successful iteration have size $\leq n_i$.

2. Total work: $O(\sum_i n_i Y_i)$. 
QuickSelect analysis

Total work: \( O(\sum_i n_i Y_i) \).

We have:

1. \( n_i \leq (3/4)n_{i-1} \leq (3/4)^{i-1}n \).

2. \( Y_i \) is a random variable with geometric distribution
   Probability of \( Y_i = k \) is \( 1/2^i \).

3. \( \mathbb{E}[Y_i] = 2 \).

As such, expected work is proportional to

\[
\mathbb{E}\left[ \sum_i n_i Y_i \right] = \sum_i \mathbb{E}[n_i Y_i] \leq \sum_i \mathbb{E}\left[ (3/4)^{i-1} n Y_i \right] \\
= n \sum_i (3/4)^{i-1} \mathbb{E}[Y_i] = n \sum_{i=1} (3/4)^{i-1}2 \leq 8n.
\]
The expected running time of QuickSelect is $O(n)$. 

**Theorem**

*The expected running time of QuickSelect is $O(n)$.***