The Randomized Quicksort Algorithm

K. Subramani[1]

[1] Lane Department of Computer Science and Electrical Engineering
West Virginia University

7 February, 2012

Outline

1. The Randomized Quicksort Algorithm

The Sorting Problem

Problem Statement

Given an array **A** of n distinct integers, in the indices $A[1]$ through $A[n]$,

The Sorting Problem

Problem Statement

Given an array **A** of n distinct integers, in the indices $A[1]$ through $A[n]$, permute the elements of **A**, so that

The Sorting Problem

Problem Statement

Given an array **A** of n distinct integers, in the indices $A[1]$ through $A[n]$, permute the elements of **A**, so that $A[1] < A[2] \ldots A[n]$.

The Sorting Problem

Problem Statement

Given an array **A** of n distinct integers, in the indices $A[1]$ through $A[n]$, permute the elements of **A**, so that $A[1] < A[2] \ldots A[n]$.

Note

The assumption of distinctness simplifies the analysis.

The Sorting Problem

Problem Statement

Given an array **A** of n distinct integers, in the indices $A[1]$ through $A[n]$, permute the elements of **A**, so that $A[1] < A[2] \ldots A[n]$.

Note

The assumption of distinctness simplifies the analysis. It has no bearing on the running time.

The Partition subroutine

Function PARTITION$(\mathbf{A}, p, q)$

```
{We partition the sub-array A[p, p+1, ..., q] about A[p].}
for (i = (p+1) to q) do
    if (A[i] < A[p]) then
        Insert A[i] into bucket L.
    else
        if (A[i] > A[p]) then
            Insert A[i] into bucket U.
        end if
    end if
end for
Copy A[p] into A[(|L|+1)].
Copy the elements of L into the first |L| entries of A[p··q].
Copy A[p] into A[(|L|+1)].
Copy the elements of U into the entries of A[(|L|+2)··q].
return (|L|+1).
```

The Partition subroutine

Function PARTITION($\mathbf{A}, p, q$)

$\{$We partition the sub-array $\mathbf{A}[\mathbf{p}, \mathbf{p}+\mathbf{1}, \ldots, \mathbf{q}]$ about $A[p]$.$\}$
for $(i = (p+1)$ **to** $q)$ **do**
    if $(A[i] < A[p])$ **then**
        Insert $A[i]$ into bucket $\mathbf{L}$.
    else
        if $(A[i] > A[p])$ **then**
            Insert $A[i]$ into bucket $\mathbf{U}$.
        end if
    end if
end for
Copy $A[p]$ into $A[(|\mathbf{L}|+1)]$.
Copy the elements of $\mathbf{L}$ into the first $|\mathbf{L}|$ entries of $\mathbf{A}[p \cdot\cdot q]$.
Copy $A[p]$ into $A[(|\mathbf{L}|+1)]$.
Copy the elements of $\mathbf{U}$ into the entries of $\mathbf{A}[(|\mathbf{L}|+2) \cdot\cdot q]$.
return $(|\mathbf{L}|+1)$.

Note

Partitioning an array can be achieved in linear time.

The Quicksort Algorithm

The Quicksort Algorithm

Function QUICKSORT$(\mathbf{A},\ p,\ q)$

```
if (p ≥ q) then
    return
else
    j = PARTITION(A, p, q).
    Quicksort(A, p, j − 1).
    Quicksort(A, j + 1, q).
end if
```

The Quicksort Algorithm

Function $\textsc{Quicksort}(\mathbf{A},\ p,\ q)$

if $(p \geq q)$ **then**
    return
else
    $j = \textsc{Partition}(\mathbf{A},\ p,\ q)$.
    Quicksort($\mathbf{A}$, $p, j-1$).
    Quicksort($\mathbf{A}$, $j+1, q$).
end if

Note

The main program calls $\textsc{Quicksort}(\mathbf{A},\ 1,\ n)$.

Worst-case analysis

Analysis

What is the worst-case input for QUICKSORT()?

Worst-case analysis

Analysis

What is the worst-case input for QUICKSORT()? How many comparisons in the worst case?

Worst-case analysis

Analysis

What is the worst-case input for QUICKSORT()? How many comparisons in the worst case? $O(n^2)$.

Worst-case analysis

Analysis

What is the worst-case input for QUICKSORT()? How many comparisons in the worst case? $O(n^2)$.

Intuition for randomized case

What sort of assumptions are reasonable in analysis?

Randomized Quicksort

Randomized Quicksort

Function RANDOMIZED-QUICKSORT$(\mathbf{A},\ p,\ q)$

if $(p \geq q)$ **then**
    return
else
    Choose a number, say r, uniformly and at random from the set $\{p, p+1, \ldots, q\}$.
    Swap $A[p]$ and $A[r]$.
    $j =$ PARTITION$(\mathbf{A},\ p,\ q)$.
    Quicksort($\mathbf{A}$, $p, j-1$).
    Quicksort($\mathbf{A}$, $j+1$, q).
end if

Randomized Quicksort

Function $\textsc{Randomized-Quicksort}(\mathbf{A},\ p,\ q)$

```
if (p ≥ q) then
    return
else
    Choose a number, say r, uniformly and at random from the set {p, p+1, ..., q}.
    Swap A[p] and A[r].
    j = PARTITION(A, p, q).
    Quicksort(A, p, j−1).
    Quicksort(A, j+1, q).
end if
```

Note

Worst case running time?

Randomized Quicksort

Function RANDOMIZED-QUICKSORT($\mathbf{A}$, p, q)

if $(p \geq q)$ **then**
    return
else
    Choose a number, say r, uniformly and at random from the set $\{p, p+1, \ldots, q\}$.
    Swap $A[p]$ and $A[r]$.
    $j =$ PARTITION($\mathbf{A}$, p, q).
    Quicksort($\mathbf{A}$, p, $j-1$).
    Quicksort($\mathbf{A}$, $j+1$, q).
end if

Note

Worst case running time? $O(n^2)$*!*

Randomized Quicksort

Function RANDOMIZED-QUICKSORT$(\mathbf{A},\ p,\ q)$

if $(p \geq q)$ **then**

  return

else

  Choose a number, say r, uniformly and at random from the set $\{p, p+1, \ldots, q\}$.

  Swap $A[p]$ and $A[r]$.

  $j =$ PARTITION$(\mathbf{A},\ p,\ q)$.

  Quicksort($\mathbf{A}$, $p, j-1$).

  Quicksort($\mathbf{A}$, $j+1$, q).

end if

Note

Worst case running time? $O(n^2)$*! However, for a randomized algorithm we are not interested in worst-case running time, but in expected running time.*

Decision Tree Analysis

Decision Tree Analysis

Decision Tree

The operation of RANDOMIZED QUICKSORT() can be thought of as a binary tree, say T, with a pivot being chosen at each internal node.

Decision Tree Analysis

Decision Tree

The operation of RANDOMIZED QUICKSORT() can be thought of as a binary tree, say T, with a pivot being chosen at each internal node. The elements in the node which are less than the pivot are shunted to the left subtree and the rest of the elements (excluding the pivot) are shunted to the right subtree.

Decision Tree Analysis

Decision Tree

The operation of RANDOMIZED QUICKSORT() can be thought of as a binary tree, say T, with a pivot being chosen at each internal node. The elements in the node which are less than the pivot are shunted to the left subtree and the rest of the elements (excluding the pivot) are shunted to the right subtree. An in-order traversal of T focusing on the pivots, gives the sorted order.

Decision Tree Analysis

Decision Tree

The operation of RANDOMIZED QUICKSORT() can be thought of as a binary tree, say T, with a pivot being chosen at each internal node. The elements in the node which are less than the pivot are shunted to the left subtree and the rest of the elements (excluding the pivot) are shunted to the right subtree. An in-order traversal of T focusing on the pivots, gives the sorted order. What is the work done at each level of the tree?

Decision Tree Analysis

Decision Tree

The operation of RANDOMIZED QUICKSORT() can be thought of as a binary tree, say T, with a pivot being chosen at each internal node. The elements in the node which are less than the pivot are shunted to the left subtree and the rest of the elements (excluding the pivot) are shunted to the right subtree. An in-order traversal of T focusing on the pivots, gives the sorted order. What is the work done at each level of the tree? $O(n)$.

Decision Tree Analysis

Decision Tree

The operation of RANDOMIZED QUICKSORT() can be thought of as a binary tree, say T, with a pivot being chosen at each internal node. The elements in the node which are less than the pivot are shunted to the left subtree and the rest of the elements (excluding the pivot) are shunted to the right subtree. An in-order traversal of T focusing on the pivots, gives the sorted order. What is the work done at each level of the tree? $O(n)$. Let h denote the height of T.

Decision Tree Analysis

Decision Tree

The operation of RANDOMIZED QUICKSORT() can be thought of as a binary tree, say T, with a pivot being chosen at each internal node. The elements in the node which are less than the pivot are shunted to the left subtree and the rest of the elements (excluding the pivot) are shunted to the right subtree. An in-order traversal of T focusing on the pivots, gives the sorted order. What is the work done at each level of the tree? $O(n)$. Let h denote the height of T. Observe that h is a random variable and we are interested in its expected value.

Decision Tree Analysis

Decision Tree

The operation of RANDOMIZED QUICKSORT() can be thought of as a binary tree, say T, with a pivot being chosen at each internal node. The elements in the node which are less than the pivot are shunted to the left subtree and the rest of the elements (excluding the pivot) are shunted to the right subtree. An in-order traversal of T focusing on the pivots, gives the sorted order. What is the work done at each level of the tree? $O(n)$. Let h denote the height of T. Observe that h is a random variable and we are interested in its expected value.

The rank of an element of **A** is its position in **A**, when **A** has been sorted.

Decision Tree Analysis

Decision Tree

The operation of RANDOMIZED QUICKSORT() can be thought of as a binary tree, say T, with a pivot being chosen at each internal node. The elements in the node which are less than the pivot are shunted to the left subtree and the rest of the elements (excluding the pivot) are shunted to the right subtree. An in-order traversal of T focusing on the pivots, gives the sorted order. What is the work done at each level of the tree? $O(n)$. Let h denote the height of T. Observe that h is a random variable and we are interested in its expected value.

The rank of an element of **A** is its position in **A**, when **A** has been sorted.

When you pick an element at random, what is the probability that the rank of the element chosen is between $\frac{1}{4} \cdot n$ and $\frac{3}{4} \cdot n$, where n is the number of elements in the array?

Decision Tree Analysis

Decision Tree

The operation of RANDOMIZED QUICKSORT() can be thought of as a binary tree, say T, with a pivot being chosen at each internal node. The elements in the node which are less than the pivot are shunted to the left subtree and the rest of the elements (excluding the pivot) are shunted to the right subtree. An in-order traversal of T focusing on the pivots, gives the sorted order. What is the work done at each level of the tree? $O(n)$. Let h denote the height of T. Observe that h is a random variable and we are interested in its expected value.

The rank of an element of **A** is its position in **A**, when **A** has been sorted.

When you pick an element at random, what is the probability that the rank of the element chosen is between $\frac{1}{4} \cdot n$ and $\frac{3}{4} \cdot n$, where n is the number of elements in the array? $\frac{1}{2}$.

Consider the tree T.

Decision Tree Analysis

Decision Tree

The operation of RANDOMIZED QUICKSORT() can be thought of as a binary tree, say T, with a pivot being chosen at each internal node. The elements in the node which are less than the pivot are shunted to the left subtree and the rest of the elements (excluding the pivot) are shunted to the right subtree. An in-order traversal of T focusing on the pivots, gives the sorted order. What is the work done at each level of the tree? $O(n)$. Let h denote the height of T. Observe that h is a random variable and we are interested in its expected value.

The rank of an element of **A** is its position in **A**, when **A** has been sorted.

When you pick an element at random, what is the probability that the rank of the element chosen is between $\frac{1}{4} \cdot n$ and $\frac{3}{4} \cdot n$, where n is the number of elements in the array? $\frac{1}{2}$.

Consider the tree T. We define an internal node o of the tree to be **good**, if both its children have at most $\frac{3}{4} \cdot |o|$ nodes, where $|o|$ denotes the number of elements in the node o.

Decision Tree Analysis

Decision Tree

The operation of RANDOMIZED QUICKSORT() can be thought of as a binary tree, say T, with a pivot being chosen at each internal node. The elements in the node which are less than the pivot are shunted to the left subtree and the rest of the elements (excluding the pivot) are shunted to the right subtree. An in-order traversal of T focusing on the pivots, gives the sorted order. What is the work done at each level of the tree? $O(n)$. Let h denote the height of T. Observe that h is a random variable and we are interested in its expected value.

The rank of an element of **A** is its position in **A**, when **A** has been sorted.

When you pick an element at random, what is the probability that the rank of the element chosen is between $\frac{1}{4} \cdot n$ and $\frac{3}{4} \cdot n$, where n is the number of elements in the array? $\frac{1}{2}$.

Consider the tree T. We define an internal node o of the tree to be **good**, if both its children have at most $\frac{3}{4} \cdot |o|$ nodes, where $|o|$ denotes the number of elements in the node o. Given an intenal node o of T, what is the probability that it is **good**?

Decision Tree Analysis

Decision Tree

The operation of RANDOMIZED QUICKSORT() can be thought of as a binary tree, say T, with a pivot being chosen at each internal node. The elements in the node which are less than the pivot are shunted to the left subtree and the rest of the elements (excluding the pivot) are shunted to the right subtree. An in-order traversal of T focusing on the pivots, gives the sorted order. What is the work done at each level of the tree? $O(n)$. Let h denote the height of T. Observe that h is a random variable and we are interested in its expected value.

The rank of an element of **A** is its position in **A**, when **A** has been sorted.

When you pick an element at random, what is the probability that the rank of the element chosen is between $\frac{1}{4} \cdot n$ and $\frac{3}{4} \cdot n$, where n is the number of elements in the array? $\frac{1}{2}$.

Consider the tree T. We define an internal node o of the tree to be **good**, if both its children have at most $\frac{3}{4} \cdot |o|$ nodes, where $|o|$ denotes the number of elements in the node o. Given an intenal node o of T, what is the probability that it is **good**? At least $\frac{1}{2}$!

Decision Tree Analysis

Decision Tree

The operation of RANDOMIZED QUICKSORT() can be thought of as a binary tree, say T, with a pivot being chosen at each internal node. The elements in the node which are less than the pivot are shunted to the left subtree and the rest of the elements (excluding the pivot) are shunted to the right subtree. An in-order traversal of T focusing on the pivots, gives the sorted order. What is the work done at each level of the tree? $O(n)$. Let h denote the height of T. Observe that h is a random variable and we are interested in its expected value.
The rank of an element of **A** is its position in **A**, when **A** has been sorted.
When you pick an element at random, what is the probability that the rank of the element chosen is between $\frac{1}{4} \cdot n$ and $\frac{3}{4} \cdot n$, where n is the number of elements in the array? $\frac{1}{2}$.
Consider the tree T. We define an internal node o of the tree to be **good**, if both its children have at most $\frac{3}{4} \cdot |o|$ nodes, where $|o|$ denotes the number of elements in the node o. Given an intenal node o of T, what is the probability that it is **good**? At least $\frac{1}{2}$!
Consider a root to leaf path in T.

Decision Tree Analysis

Decision Tree

The operation of RANDOMIZED QUICKSORT() can be thought of as a binary tree, say T, with a pivot being chosen at each internal node. The elements in the node which are less than the pivot are shunted to the left subtree and the rest of the elements (excluding the pivot) are shunted to the right subtree. An in-order traversal of T focusing on the pivots, gives the sorted order. What is the work done at each level of the tree? $O(n)$. Let h denote the height of T. Observe that h is a random variable and we are interested in its expected value.

The rank of an element of **A** is its position in **A**, when **A** has been sorted.

When you pick an element at random, what is the probability that the rank of the element chosen is between $\frac{1}{4} \cdot n$ and $\frac{3}{4} \cdot n$, where n is the number of elements in the array? $\frac{1}{2}$.

Consider the tree T. We define an internal node o of the tree to be **good**, if both its children have at most $\frac{3}{4} \cdot |o|$ nodes, where $|o|$ denotes the number of elements in the node o. Given an intenal node o of T, what is the probability that it is **good**? At least $\frac{1}{2}$!

Consider a root to leaf path in T. How many **good** nodes can exist on such a path?

Decision Tree Analysis

Decision Tree

The operation of RANDOMIZED QUICKSORT() can be thought of as a binary tree, say T, with a pivot being chosen at each internal node. The elements in the node which are less than the pivot are shunted to the left subtree and the rest of the elements (excluding the pivot) are shunted to the right subtree. An in-order traversal of T focusing on the pivots, gives the sorted order. What is the work done at each level of the tree? $O(n)$. Let h denote the height of T. Observe that h is a random variable and we are interested in its expected value.

The rank of an element of **A** is its position in **A**, when **A** has been sorted.

When you pick an element at random, what is the probability that the rank of the element chosen is between $\frac{1}{4} \cdot n$ and $\frac{3}{4} \cdot n$, where n is the number of elements in the array? $\frac{1}{2}$.

Consider the tree T. We define an internal node o of the tree to be **good**, if both its children have at most $\frac{3}{4} \cdot |o|$ nodes, where $|o|$ denotes the number of elements in the node o. Given an intenal node o of T, what is the probability that it is **good**? At least $\frac{1}{2}$!

Consider a root to leaf path in T. How many **good** nodes can exist on such a path? At most $r = \log_{\frac{4}{3}} n$.

Decision Tree Analysis

Decision Tree

The operation of RANDOMIZED QUICKSORT() can be thought of as a binary tree, say T, with a pivot being chosen at each internal node. The elements in the node which are less than the pivot are shunted to the left subtree and the rest of the elements (excluding the pivot) are shunted to the right subtree. An in-order traversal of T focusing on the pivots, gives the sorted order. What is the work done at each level of the tree? $O(n)$. Let h denote the height of T. Observe that h is a random variable and we are interested in its expected value.

The rank of an element of **A** is its position in **A**, when **A** has been sorted.

When you pick an element at random, what is the probability that the rank of the element chosen is between $\frac{1}{4} \cdot n$ and $\frac{3}{4} \cdot n$, where n is the number of elements in the array? $\frac{1}{2}$.

Consider the tree T. We define an internal node o of the tree to be **good**, if both its children have at most $\frac{3}{4} \cdot |o|$ nodes, where $|o|$ denotes the number of elements in the node o. Given an intenal node o of T, what is the probability that it is **good**? At least $\frac{1}{2}$!

Consider a root to leaf path in T. How many **good** nodes can exist on such a path? At most $r = \log_{\frac{4}{3}} n$. What is the *expected* number of nodes on a root to leaf path before you see r good nodes?

Decision Tree Analysis (contd.)

Lemma

Consider a coin for which the probability of "heads" turning up on a toss is p.

Decision Tree Analysis (contd.)

Lemma

Consider a coin for which the probability of "heads" turning up on a toss is p. What is the expected number of tosses to obtain k heads?

Decision Tree Analysis (contd.)

Lemma

Consider a coin for which the probability of "heads" turning up on a toss is p. What is the expected number of tosses to obtain k heads? $\frac{k}{p}$.

Decision Tree (contd.)

Decision Tree Analysis (contd.)

Lemma

Consider a coin for which the probability of "heads" turning up on a toss is p. What is the expected number of tosses to obtain k heads? $\frac{k}{p}$.

Decision Tree (contd.)

Thus the expected number of nodes on a root to leaf path is

Decision Tree Analysis (contd.)

Lemma

Consider a coin for which the probability of "heads" turning up on a toss is p. What is the expected number of tosses to obtain k heads? $\frac{k}{p}$.

Decision Tree (contd.)

Thus the expected number of nodes on a root to leaf path is $\frac{r}{\frac{1}{2}} =$

Decision Tree Analysis (contd.)

Lemma

Consider a coin for which the probability of "heads" turning up on a toss is p. What is the expected number of tosses to obtain k heads? $\frac{k}{p}$.

Decision Tree (contd.)

Thus the expected number of nodes on a root to leaf path is $\frac{r}{\frac{1}{2}} = 2 \cdot r =$

Decision Tree Analysis (contd.)

Lemma

Consider a coin for which the probability of "heads" turning up on a toss is p. What is the expected number of tosses to obtain k heads? $\frac{k}{p}$.

Decision Tree (contd.)

Thus the expected number of nodes on a root to leaf path is $\frac{r}{\frac{1}{2}} = 2 \cdot r = 2 \cdot \log_{\frac{4}{3}} n$.

Decision Tree Analysis (contd.)

Lemma

Consider a coin for which the probability of "heads" turning up on a toss is p. What is the expected number of tosses to obtain k heads? $\frac{k}{p}$.

Decision Tree (contd.)

Thus the expected number of nodes on a root to leaf path is $\frac{r}{\frac{1}{2}} = 2 \cdot r = 2 \cdot \log_{\frac{4}{3}} n$. However, this is the expected height of T, i.e., $E[h]$.

Decision Tree Analysis (contd.)

Lemma

Consider a coin for which the probability of "heads" turning up on a toss is p. What is the expected number of tosses to obtain k heads? $\frac{k}{p}$.

Decision Tree (contd.)

Thus the expected number of nodes on a root to leaf path is $\frac{r}{\frac{1}{2}} = 2 \cdot r = 2 \cdot \log_{\frac{4}{3}} n$. However, this is the expected height of T, i.e., $E[h]$. Therefore, the expected work undertaken by the algorithm

Decision Tree Analysis (contd.)

Lemma

Consider a coin for which the probability of "heads" turning up on a toss is p. What is the expected number of tosses to obtain k heads? $\frac{k}{p}$.

Decision Tree (contd.)

Thus the expected number of nodes on a root to leaf path is $\frac{r}{\frac{1}{2}} = 2 \cdot r = 2 \cdot \log_{\frac{4}{3}} n$. However, this is the expected height of T, i.e., $E[h]$. Therefore, the expected work undertaken by the algorithm

$$E[h] \times \text{work done per level} \quad = \quad O(n \cdot \log n).$$

Indicator Variable Analysis

Indicator Variable Analysis

Definition

A random variable is an indicator variable, if it assumes the value 1, for the occurrence of some event,

Indicator Variable Analysis

Definition

A random variable is an indicator variable, if it assumes the value 1, for the occurrence of some event, and 0 otherwise.

Indicator Variable Analysis

Definition

A random variable is an indicator variable, if it assumes the value 1, for the occurrence of some event, and 0 otherwise.

Note

We recall that the rank of an array element is its position in the sorted array.

Indicator Variable Analysis

Definition

A random variable is an indicator variable, if it assumes the value 1, for the occurrence of some event, and 0 otherwise.

Note

We recall that the rank of an array element is its position in the sorted array. Every element of **A** *has a unique rank in the set* $\{1, 2, \ldots, n\}$.

Indicator Variable Analysis

Definition

A random variable is an indicator variable, if it assumes the value 1, for the occurrence of some event, and 0 otherwise.

Note

We recall that the rank of an array element is its position in the sorted array. Every element of **A** *has a unique rank in the set* $\{1, 2, \ldots, n\}$.

Analysis

Let $S(i)$ denote the element in **A**, whose rank i.

Indicator Variable Analysis

Definition

A random variable is an indicator variable, if it assumes the value 1, for the occurrence of some event, and 0 otherwise.

Note

We recall that the rank of an array element is its position in the sorted array. Every element of **A** *has a unique rank in the set* $\{1, 2, \ldots, n\}$.

Analysis

Let $S(i)$ denote the element in **A**, whose rank i. We wish to compute the number of comparisons between $A[i]$ and the other elements of **A**, for each $i = 1, 2 \ldots n$.

Indicator Variable Analysis

Definition

A random variable is an indicator variable, if it assumes the value 1, for the occurrence of some event, and 0 otherwise.

Note

We recall that the rank of an array element is its position in the sorted array. Every element of **A** *has a unique rank in the set* $\{1, 2, \ldots, n\}$.

Analysis

Let $S(i)$ denote the element in **A**, whose rank i. We wish to compute the number of comparisons between $A[i]$ and the other elements of **A**, for each $i = 1, 2 \ldots n$. Instead, we will compute the number of comparisons between $S(i)$ and the elements of other ranks, for each $i = 1, 2, \ldots, n$.

Indicator Variable Analysis

Definition

A random variable is an indicator variable, if it assumes the value 1, for the occurrence of some event, and 0 otherwise.

Note

We recall that the rank of an array element is its position in the sorted array. Every element of **A** *has a unique rank in the set* $\{1, 2, \ldots, n\}$.

Analysis

Let $S(i)$ denote the element in **A**, whose rank i. We wish to compute the number of comparisons between $A[i]$ and the other elements of **A**, for each $i = 1, 2 \ldots n$. Instead, we will compute the number of comparisons between $S(i)$ and the elements of other ranks, for each $i = 1, 2, \ldots, n$. Are the two computations equivalent?

Indicator Variable Analysis (contd.)

Indicator Variable Analysis (contd.)

Analysis (contd.)

Let X_{ij} denote an indicator random variable, defined as follows:

$$X_{ij} = \begin{cases} 1, & \text{if S(i) and S(j) are compared during the course of the algorithm} \\ 0, & \text{otherwise} \end{cases}$$

Let X denote the total number of comparisons made by the algorithm.

Indicator Variable Analysis (contd.)

Analysis (contd.)

Let X_{ij} denote an indicator random variable, defined as follows:

$$X_{ij} = \begin{cases} 1, & \text{if S(i) and S(j) are compared during the course of the algorithm} \\ 0, & \text{otherwise} \end{cases}$$

Let X denote the total number of comparisons made by the algorithm. Clearly,

$$X \quad =$$

Indicator Variable Analysis (contd.)

Analysis (contd.)

Let X_{ij} denote an indicator random variable, defined as follows:

$$X_{ij} = \begin{cases} 1, & \text{if S(i) and S(j) are compared during the course of the algorithm} \\ 0, & \text{otherwise} \end{cases}$$

Let X denote the total number of comparisons made by the algorithm. Clearly,

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

Indicator Variable Analysis (contd.)

Analysis (contd.)

Let X_{ij} denote an indicator random variable, defined as follows:

$$X_{ij} = \begin{cases} 1, & \text{if S(i) and S(j) are compared during the course of the algorithm} \\ 0, & \text{otherwise} \end{cases}$$

Let X denote the total number of comparisons made by the algorithm. Clearly,

$$\begin{aligned} X &= \sum_{i=1}^{n-1} \sum_{j>i} X_{ij} \\ &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij} \end{aligned}$$

Indicator Variable Analysis (contd.)

Analysis (contd.)

Let X_{ij} denote an indicator random variable, defined as follows:

$$X_{ij} = \begin{cases} 1, & \text{if S(i) and S(j) are compared during the course of the algorithm} \\ 0, & \text{otherwise} \end{cases}$$

Let X denote the total number of comparisons made by the algorithm. Clearly,

$$\begin{aligned} X &= \sum_{i=1}^{n-1} \sum_{j>i} X_{ij} \\ &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij} \end{aligned}$$

How to compute X?

Indicator Variable Analysis (contd.)

Analysis (contd.)

Let X_{ij} denote an indicator random variable, defined as follows:

$$X_{ij} = \begin{cases} 1, & \text{if S(i) and S(j) are compared during the course of the algorithm} \\ 0, & \text{otherwise} \end{cases}$$

Let X denote the total number of comparisons made by the algorithm. Clearly,

$$\begin{aligned} X &= \sum_{i=1}^{n-1} \sum_{j>i} X_{ij} \\ &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij} \end{aligned}$$

How to compute X? We are not interested in X, but in $E[X]$!

Indicator Variable Analysis (contd.)

Analysis (contd.)

Let X_{ij} denote an indicator random variable, defined as follows:

$$X_{ij} = \begin{cases} 1, & \text{if S(i) and S(j) are compared during the course of the algorithm} \\ 0, & \text{otherwise} \end{cases}$$

Let X denote the total number of comparisons made by the algorithm. Clearly,

$$\begin{aligned} X &= \sum_{i=1}^{n-1} \sum_{j>i} X_{ij} \\ &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij} \end{aligned}$$

How to compute X? We are not interested in X, but in $E[X]$! Observe that,

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

Indicator Variable Analysis (contd.)

Indicator Variable Analysis (contd.)

Analysis (contd.)

$$E[X] \quad =$$

Indicator Variable Analysis (contd.)

Analysis (contd.)

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

Indicator Variable Analysis (contd.)

Analysis (contd.)

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

Let p_{ij} denote the probability that $S(i)$ and $S(j)$ are compared.

Indicator Variable Analysis (contd.)

Analysis (contd.)

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

Let p_{ij} denote the probability that $S(i)$ and $S(j)$ are compared. Clearly, $E[X_{ij}] =$

Indicator Variable Analysis (contd.)

Analysis (contd.)

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

Let p_{ij} denote the probability that $S(i)$ and $S(j)$ are compared. Clearly, $E[X_{ij}] = p_{ij}$.

Indicator Variable Analysis (contd.)

Analysis (contd.)

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

Let p_{ij} denote the probability that $S(i)$ and $S(j)$ are compared. Clearly, $E[X_{ij}] = p_{ij}$. How to compute p_{ij}?

Indicator Variable Analysis (contd.)

Analysis (contd.)

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

Let p_{ij} denote the probability that $S(i)$ and $S(j)$ are compared. Clearly, $E[X_{ij}] = p_{ij}$. How to compute p_{ij}? Let $S_{ij} = \{S(i), S(i+1), \ldots S(j)\}$.

Indicator Variable Analysis (contd.)

Analysis (contd.)

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

Let p_{ij} denote the probability that $S(i)$ and $S(j)$ are compared. Clearly, $E[X_{ij}] = p_{ij}$. How to compute p_{ij}? Let $S_{ij} = \{S(i), S(i+1), \ldots S(j)\}$. $S(i)$ and $S(j)$ will be compared **only if**, either one of them is picked before the other elements in S_{ij}!

Indicator Variable Analysis (contd.)

Analysis (contd.)

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

Let p_{ij} denote the probability that $S(i)$ and $S(j)$ are compared. Clearly, $E[X_{ij}] = p_{ij}$. How to compute p_{ij}? Let $S_{ij} = \{S(i), S(i+1), \ldots S(j)\}$. $S(i)$ and $S(j)$ will be compared **only if**, either one of them is picked before the other elements in S_{ij}! Since all choices are made uniformly and at random, the probability of either $S(i)$ or $S(j)$ being picked before the other elements in S_{ij} is exactly

Indicator Variable Analysis (contd.)

Analysis (contd.)

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

Let p_{ij} denote the probability that $S(i)$ and $S(j)$ are compared. Clearly, $E[X_{ij}] = p_{ij}$. How to compute p_{ij}? Let $S_{ij} = \{S(i), S(i+1), \ldots S(j)\}$. $S(i)$ and $S(j)$ will be compared **only if**, either one of them is picked before the other elements in S_{ij}! Since all choices are made uniformly and at random, the probability of either $S(i)$ or $S(j)$ being picked before the other elements in S_{ij} is exactly $\frac{2}{j-i+1}$.

Indicator Variable Analysis (contd.)

Analysis (contd.)

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

Let p_{ij} denote the probability that $S(i)$ and $S(j)$ are compared. Clearly, $E[X_{ij}] = p_{ij}$. How to compute p_{ij}? Let $S_{ij} = \{S(i), S(i+1), \ldots S(j)\}$. $S(i)$ and $S(j)$ will be compared **only if**, either one of them is picked before the other elements in S_{ij}! Since all choices are made uniformly and at random, the probability of either $S(i)$ or $S(j)$ being picked before the other elements in S_{ij} is exactly $\frac{2}{j-i+1}$.
Therefore,

Indicator Variable Analysis (contd.)

Analysis (contd.)

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

Let p_{ij} denote the probability that $S(i)$ and $S(j)$ are compared. Clearly, $E[X_{ij}] = p_{ij}$. How to compute p_{ij}? Let $S_{ij} = \{S(i), S(i+1), \ldots S(j)\}$. $S(i)$ and $S(j)$ will be compared **only if**, either one of them is picked before the other elements in S_{ij}! Since all choices are made uniformly and at random, the probability of either $S(i)$ or $S(j)$ being picked before the other elements in S_{ij} is exactly $\frac{2}{j-i+1}$.
Therefore,

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

Indicator Variable Analysis (contd.)

Analysis (contd.)

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

Let p_{ij} denote the probability that $S(i)$ and $S(j)$ are compared. Clearly, $E[X_{ij}] = p_{ij}$. How to compute p_{ij}? Let $S_{ij} = \{S(i), S(i+1), \ldots S(j)\}$. $S(i)$ and $S(j)$ will be compared **only if**, either one of them is picked before the other elements in S_{ij}! Since all choices are made uniformly and at random, the probability of either $S(i)$ or $S(j)$ being picked before the other elements in S_{ij} is exactly $\frac{2}{j-i+1}$.
Therefore,

$$\begin{aligned} E[X] &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} E[X_{ij}] \\ &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1} \end{aligned}$$

Indicator Variable Analysis (contd.)

Analysis (contd.)

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

Let p_{ij} denote the probability that $S(i)$ and $S(j)$ are compared. Clearly, $E[X_{ij}] = p_{ij}$. How to compute p_{ij}? Let $S_{ij} = \{S(i), S(i+1), \ldots S(j)\}$. $S(i)$ and $S(j)$ will be compared **only if**, either one of them is picked before the other elements in S_{ij}! Since all choices are made uniformly and at random, the probability of either $S(i)$ or $S(j)$ being picked before the other elements in S_{ij} is exactly $\frac{2}{j-i+1}$.
Therefore,

$$\begin{aligned} E[X] &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} E[X_{ij}] \\ &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1} \\ &= \sum_{i=1}^{n-1} \sum_{k=2}^{n-i+1} \frac{2}{k} \end{aligned}$$

Indicator Variable Analysis (contd.)

Indicator Variable Analysis (contd.)

Analysis (contd.)

$$E[X] \leq \sum_{i=1}^{n} \sum_{k=1}^{n} \frac{2}{k}$$

Indicator Variable Analysis (contd.)

Analysis (contd.)

$$
\begin{aligned}
E[X] &\leq \sum_{i=1}^{n}\sum_{k=1}^{n}\frac{2}{k} \\
&= 2\cdot\sum_{i=1}^{n}\sum_{k=1}^{n}\frac{1}{k}
\end{aligned}
$$

Indicator Variable Analysis (contd.)

Analysis (contd.)

$$
\begin{aligned}
E[X] &\leq \sum_{i=1}^{n} \sum_{k=1}^{n} \frac{2}{k} \\
&= 2 \cdot \sum_{i=1}^{n} \sum_{k=1}^{n} \frac{1}{k} \\
&= 2 \cdot \sum_{i=1}^{n} H_n
\end{aligned}
$$

Indicator Variable Analysis (contd.)

Analysis (contd.)

$$
\begin{aligned}
E[X] &\leq \sum_{i=1}^{n}\sum_{k=1}^{n}\frac{2}{k} \\
&= 2\cdot\sum_{i=1}^{n}\sum_{k=1}^{n}\frac{1}{k} \\
&= 2\cdot\sum_{i=1}^{n} H_n \\
&= 2\cdot n\cdot H_n
\end{aligned}
$$

Indicator Variable Analysis (contd.)

Analysis (contd.)

$$
\begin{aligned}
E[X] &\leq \sum_{i=1}^{n}\sum_{k=1}^{n}\frac{2}{k} \\
&= 2\cdot\sum_{i=1}^{n}\sum_{k=1}^{n}\frac{1}{k} \\
&= 2\cdot\sum_{i=1}^{n} H_n \\
&= 2\cdot n\cdot H_n \\
&\in O(n\cdot\log n)
\end{aligned}
$$