Week 6 Tutorial

CSCI2100 Teaching Team 2021

Department of Computer Science and Engineering

The Chinese University of HongKong
Pivot Selection

**Input:** An array $A$ of $n$ integers in arbitrary order.

**Output:** An element in $A$ whose rank is between $\frac{n}{10}$ and $\frac{9n}{10}$.

Example:

```
2 3 1 4 5 9 7 6 10 8
```

$A$

Valid answers: any number from 2 to 9.
Pivot Selection

**Algorithm**

1. Randomly pick an integer $v$ from $A$; call $v$ the pivot.
2. Get the rank $r$ of $v$.
3. If $r$ is not in $[n/10, 9n/10]$, repeat from 1.
4. Otherwise, output $v$. 
Cost Analysis

How many times do we have to repeat Step 1 and 2?

Each run finds a valid answer $v$ with probability $4/5$. Thus, we need to repeat $5/4$ times in expectation.

Hence, our algorithm finishes in $O(n)$ expected time.

**Think:** If we use the pivot picked in the above manner for $k$-selection, what is the expected cost of the $k$-selection algorithm discussed in the lecture?
Pivot Selection

Think: what if

Input: An array $A$ of $n$ integers in arbitrary order.
Output: An element in $A$ whose rank is between $0.4999n$ and $0.5n$?
The next few slides will introduce you to some basic ideas behind generating a random number. As you will see, all we need is the ability to generate a random bit.
Coin Game 1

Given a fair coin, how do you generate a number from 1 to 4 uniformly at random?
Coin Game 1

Given a fair coin, how do you generate a number from 1 to 4 uniformly at random?

**Solution:** Flip the coin twice. Assign numbers as follows:

- (Head, Head): 1
- (Head, Tail): 2
- (Tail, Head): 3
- (Tail, Tail): 4
Coin Game 2

Given a fair coin, how do you generate a number from 1 to 3 uniformly at random?

**Hint:** Use the previous algorithm as a black box.
Given a fair coin, how do you generate a number from 1 to 3 uniformly at random?

**Solution:** Run the algorithm in Coin Game 1. If the algorithm returns 4, ignore it and run again.

**Cost:** The number of repeats is $O(1)$. 
Given a fair coin, how do you generate a number from 1 to $n$ uniformly at random?

Solution: See a regular exercise.

Example: $n = 37$.

1. Generate a number $x$ in [1, 64] uniformly at random.
2. If $x > 37$, repeat step 1.

The number of repeats is $O(1)$. 
In the next part of the tutorial, we will show how to sort a multi-set.
Sorting a Multi-Set

So far we have assumed the input to sorting is a set $S$ of integers.

What if we want to sort a multi-set $A$, i.e. a collection of integers which may contain duplicates?

$\begin{array}{cccccccc}
2 & 3 & 7 & 1 & 4 & 5 & 5 & 6 & 2 & 8 & 6 & 7 \\
\end{array}$

$A$
Merge Sort

1. Sort the first half of the array $A$.
2. Sort the second half of the array $A$.
3. Consider both subproblems solved and merge the two halves of the array into the final sorted sequence.

We only need to modify Step 3.
Merging

At the beginning, set \( i = j = 1 \).

Repeat until \( i > n/2 \) or \( j > n/2 \):

1. If \( A_1[i] \) (i.e., the \( i \)-th integer of \( A_1 \)) is smaller or equal to \( A_2[j] \), append \( A_1[i] \) to \( A \), and increase \( i \) by 1.

2. Otherwise, append \( A_2[j] \) to \( A \), and increase \( j \) by 1.
Next, we will show how to break ties using composite keys. With this technique, we can turn any comparison-based algorithm designed for sorting sets into another algorithm for sorting multisets.
Composite Keys

1. Convert every integer in $A$ to a key-id pair.
2. Break tie by comparing the ids.
   - $(a_1, b_1) < (a_2, b_2) \iff a_1 < a_2 \text{ or } a_1 = a_2, b_1 < b_2$.

Example: Convert the array $A$.

\[
\begin{array}{cccccccccccc}
2 & 3 & 7 & 1 & 4 & 5 & 5 & 6 & 2 & 8 & 6 & 7 \\
\end{array}
\]

\[
\begin{array}{cccccccccccc}
(2, 1) & (3, 2) & (7, 3) & (1, 4) & (4, 5) & (5, 6) & (5, 7) & (6, 8) & (2, 9) & (2, 10) & (8, 11) & (7, 12) \\
\end{array}
\]
Quick Sort Input: An array $A = (5, 9, 3, 10, 26, 37, 14, 12)$.

What is the probability that the algorithm compares the numbers 3 and 37?

Observations:
- Eventually, every integer will be selected as a pivot.
- 3 and 37 are not compared, if any integer between them gets selected as a pivot before 3 and 37.

Example: If 10 is the first pivot, then 3 and 37 will be separated and will not be compared in the rest of the algorithm.
Bonus: Quick Sort Exercise

**Solution:** 3 and 37 are compared if and only if either one is the first pivot among all integers in $A$.

The probability is $\frac{2}{|A|} = \frac{1}{4}$. 
**Bonus: Quick Sort Exercise**

**Quick Sort Input:** An array \( A = (5, 9, 3, 10, 26, 37, 14, 12) \).

A more challenging problem:

What is the probability that 3 is compared with 14 in the algorithm?

**Solution:** 3 and 14 are compared if and only if either one is the first pivot among 3, 5, 9, 10, 12, 14.

The probability is \( \frac{2}{6} = \frac{1}{3} \). (**think:** why?)