Lecture 7 - Substitution Method

Fall 2025, Korea University

Instructor: Gabin An (gabin_an@korea.ac.kr)

Course Outline (Before Midterm)

- Part 1: Basics
  - ~~Divide and Conquer (w/ Integer Multiplication)~~ ✅
  - ~~Basic Sorting Algorithms (Insertion Sort & Merge Sort)~~ ✅
  - ~~Asymptotic Analysis (Big-O, Big-Theta, Big-Omega)~~ ✅
  - ~~Solving Recurrences Using Master Method~~ ✅
- Part 2: Advanced Selection and Sorting
  - ~~Median and Selection Algorithm~~ 👈
  - **Solving Recurrences Using Substitution Method** 👈
  - Quick Sort, Counting Sort, Radix Sort
- Part 3: Data Structures
  - Heaps, Binary Search Trees, Balanced BSTs

Review: Selection Problem: Finding the k-th Smallest Element

> Input: An unsorted array of size n (without duplicates)
>
> Output: Return the k-th smallest element ($1 \leq k \leq n$)

Example

Array: `[7, 2, 1, 8, 6, 3, 5, 4]` ($n = 8$)

- *k* = *1* → 1st smallest = `1`
- *k* = *3* → 3rd smallest = `3`
- *k* = *8* → 8th smallest = `8`

🔍 Example: `Select([7, 2, 1, 8, 6, 3, 5, 4], k=2)`

- Pivot = `6`

Case 1: `k=2`

Case 2: `k=6`

Case 3: `k=8`

Selection algorithm

```
def select(A, k):
    assert 1 <= k <= len(A)
    if len(A) == 1:
        return A[0]
    p = choose_pivot(A) # ⭐⭐⭐⭐⭐⭐⭐
    A_less = [x for x in A if x < p]
    A_greater = [x for x in A if x > p]
    if len(A_less) == k - 1:
        return p
    elif len(A_less) > k - 1:
        return select(A_less, k)
    else:
        return select(A_greater, k - len(A_less) - 1)
```

- The pivot only affects **runtime**, not correctness.

Median of Medians: `choose_pivot` Algorithm

```python
def choose_pivot(A):
    # Base case: if small enough, just sort and return median
    if len(A) <= 5:
        return sorted(A)[len(A) // 2]

    # Step 1: Divide A into groups of 5
    groups = [A[i:i+5] for i in range(0, len(A), 5)]

    # Step 2: Sort each group and collect their medians
    medians = [sorted(group)[len(group) // 2] for group in groups]

    # Step 3: Recursively find the median of the medians
    return select(medians, k=(len(medians)+1)//2)
```

```python
A = [13, 5, 2, 8, 9, 4, 7, 1, 6, 3, 10, 12, 11, 15, 14]
```

The 30-70 Lemma

> For every input array of length $n \geq 2$, the subarray passed to the selection recursive call has length at most $\frac{7}{10}n$.

Suppose that n is a multiple of 5, and $g = n/5$ (i.e., # groups). At least $\left\lceil \frac{g}{2} \right\rceil - 1$ medians are **less than** p, which means at least 3 elements in those $\left\lceil \frac{g}{2} \right\rceil - 1$ groups are less than p. Therefore, in A, **at least** $3(\left\lceil \frac{g}{2} \right\rceil - 1) + 2$ elements are less than p.

$$\begin{aligned} |A_{greater}| &= \text{\# of elements greater than } p \\ &\leq \text{\# total elements except pivot} - \text{\# of elements less than } p \\ &= (n-1) - 3 \cdot \left(\left\lceil \frac{g}{2} \right\rceil - 1 \right) - 2 \\ &= n - 3 \left\lceil \frac{n}{10} \right\rceil \leq n - 3 \cdot \frac{n}{10} \;\; = \frac{7}{10}n \quad \blacksquare \end{aligned}$$

Running Time Analysis of `select`

We want to analyze the runtime of `select(A, k)` when `choose_pivot` uses Median of Medians. Let $T(n)$ be the time to run `select` on an input of size n.

We split the analysis into three parts:

1. **Divide into groups of 5 and find their medians**: $O(n)$ (Sort $\frac{n}{5}$ subarrays with size 5)
2. **Recursively find median of medians**: $T(\frac{n}{5})$
3. **Partition around pivot**: Worst case, the pivot splits into:
   - $\frac{7}{10}n$ on the larger side
   - So next recursive call is at most $T(\frac{7n}{10})$

✍️ **Final Recurrence**: $T(n) \leq T(\frac{n}{5}) + T(\frac{7n}{10}) + cn$

Solving the Recurrence using the Substitution Method

Final Recurrence of `select`: $T(n) \leq T(\frac{n}{5}) + T(\frac{7n}{10}) + cn$

- How do we prove that $T(n) = O(n)$?
- We can't apply the Master Method because subproblems are of different sizes.

Let's use the **Substitution Method** (also called the **GUESS-and-VERIFY** method)!

> **Final Recurrence**: $T(n) = T(\frac{n}{5}) + T(\frac{7n}{10}) + O(n) \leq T(\frac{n}{5}) + T(\frac{7n}{10}) + cn$

To show $T(n) = O(n)$, we want to prove: $T(n) \leq c'n \quad \text{for some constant } c'$

Base Case

Assume for small $n = 1$, we have $T(n) = O(1)$

We want:

$$T(1) \leq c' \Rightarrow c' \geq T(1)$$

This holds if $c' \geq 1$.

Inductive Hypothesis

For all smaller inputs $n < k$, the recurrence holds: $T(n) \leq c'n$

Inductive Step

$T(k) = T(\frac{k}{5}) + T(\frac{7k}{10}) + O(k)$

Inductive Hypothesis

For all smaller inputs $n < k$, the recurrence holds: $T(n) \le c'n$

Inductive Step

$$T(k) = T(\frac{k}{5}) + T(\frac{7k}{10}) + ck \le c' \cdot \frac{k}{5} + c' \cdot \frac{7k}{10} + ck = c' \cdot \frac{9k}{10} + ck$$

(⭐ *Here, we substitute the running time with **the inductive hypothesis***)

To satisfy $T(k) \le c'k$, we want:

$$c' \cdot \frac{9k}{10} + ck \le c'k \quad \Rightarrow \quad c' \ge 10c$$

Conclusion

If we choose $c' = \max\{1, 10c\}$ (which is a constant factor), then:

$$T(n) \le c'n \quad \Rightarrow \quad T(n) = O(n) \quad \blacksquare$$

What Happens When We Change the Partition Size in Median of Medians?

In the Median of Medians algorithm, we normally divide the input array into groups of 5 to select a good pivot. But what happens if we change the group size to 3, 7, or 9? Does the `select` algorithm still run in worst-case $O(n)$?

Example: Groups of 3 elements

Suppose that n is a multiple of 3, and $g = n/3$ (i.e., # groups). At least $\lceil \frac{g}{2} \rceil - 1$ medians are **less than** p, which means at least **2** elements in those $\lceil \frac{g}{2} \rceil - 1$ groups are less than p. Therefore, in A, **at least** $2(\lceil \frac{g}{2} \rceil - 1) + 1$ elements are less than p.

Example: $n = 15, g = 5, p = 8$. At least 2 medians are less than 8.
Therefore, at least 2 elements in 5's and 6's groups are less than 8.

```
[  2 <  5 < 13 ] # median 5
        v
[  1 <  6 <  7 ] # median 6
        v
[  4 <  8 <  9 ] # median 8
        v
[  3 < 10 < 12 ] # median 10
        v
[ 11 < 14 < 15 ] # median 14
```

> In A, **at least** $2(\lceil \frac{g}{2} \rceil - 1) + 1$ elements are less than p.

$$
\begin{aligned}
|A_{greater}| &= \#\text{ of elements greater than } p \\
&\leq \#\text{ total elements except pivot} - \#\text{ of elements less than } p \\
&= (n-1) - 2 \cdot \left(\left\lceil \frac{g}{2} \right\rceil - 1 \right) - 1 \\
&= n - 2 \left\lceil \frac{g}{2} \right\rceil \\
&= n - 2 \left\lceil \frac{n}{6} \right\rceil \leq n - 2 \cdot \frac{n}{6} \; = \frac{2}{3} n \quad \blacksquare
\end{aligned}
$$

Approximately 33:66 Lemma this time :-) *(this is better than 30:70!! - more precise pivot)*

Running Time Analysis of `select` (with group size of 3)

We want to analyze the runtime of `select(A, k)` when `choose_pivot` uses Median of Medians. Let $T(n)$ be the time to run `select` on an input of size n.

We split the analysis into three parts:

1. **Divide into groups of 3 and find their medians**: $O(n)$ (Sort $\frac{n}{3}$ subarrays with size 3)
2. **Recursively find median of medians**: $T(\frac{n}{3})$ (⭐ previously, $T(\frac{n}{5})$)
3. **Partition around pivot**: Worst case, the pivot splits into:
    - $\frac{2}{3}n$ on the larger side
    - So next recursive call is at most $T(\frac{2n}{3})$ (⭐ previously, $T(\frac{7n}{10})$)

Q. Is this still $O(n)$?

> **Final Recurrence**: $T(n) = T(\frac{n}{3}) + T(\frac{2n}{3}) + O(n) \leq T(\frac{n}{3}) + T(\frac{2n}{3}) + cn$

To show $T(n) = O(n)$, we want to prove: $T(n) \leq c'n \quad \text{for some constant } c'$

Base Case

Assume for small $n = 1$, we have $T(n) = O(1)$

We want:

$$T(1) \leq c' \Rightarrow c' \geq T(1)$$

This holds if $c' \geq 1$.

Inductive Hypothesis

For all smaller inputs $n < k$, the recurrence holds: $T(n) \le c'n$

Inductive Step

$$T(k) = T(\frac{k}{3}) + T(\frac{2k}{3}) + ck \le c' \cdot \frac{k}{3} + c' \cdot \frac{2k}{3} + ck = c'k + ck$$

To satisfy $T(k) \le c'k$, we want:

$$c'k + ck \le c'k \quad \Rightarrow \quad ck \le 0 \quad \Rightarrow \quad c \le 0 \quad (\because k > 1)$$

However, $c > 0$. Contradiction ❌

Conclusion

$T(n)$ is not $O(n)$.

Example: Substitution Method for MergeSort (1st Try)

$$T(n) = 2T\left(\frac{n}{2}\right) + cn$$

- **Goal**: Prove $T(n) = O(n)$
- **Guess**: $T(n) \leq c'n \quad \text{for some constant } c' > 0$

Example: Substitution Method for MergeSort (with Incorrect Guess - 1st Try)

$$T(n) = 2T\left(\frac{n}{2}\right) + cn$$

- **Goal**: Prove $T(n) = O(n)$
- **Guess**: $T(n) \le c'n \quad \text{for some constant } c' > 0$
- **Inductive Hypothesis**: Assume $T(n/2) \le c' \cdot \frac{n}{2}$

$$T(n) \le 2 \cdot \left(c' \cdot \frac{n}{2}\right) + cn = c'n + cn = (c' + c)n$$

$(c' + c)n$ is **larger** than $c'n$ unless $c = 0$, which is not true. ❌

- **Conclusion**: The inequality does **not** hold. The guess $T(n) = O(n)$ is **too small**, so we must try a larger bound (e.g., $O(n \log n)$).

Example: Substitution Method for MergeSort (2nd Try)

$$T(n) = 2T\left(\frac{n}{2}\right) + cn$$

- **Goal**: Prove $T(n) = O(n \log n)$
- **Guess**: $T(n) \leq c'n \log n \quad \text{for some constant } c' > 0$

Example: Substitution Method for MergeSort (with Correct Guess - 2nd Try)

$$T(n) = 2T\left(\frac{n}{2}\right) + cn$$

- **Goal**: Prove $T(n) = O(n \log n)$
- **Guess**: $T(n) \leq c'n \log n \quad \text{for some constant } c' > 0$
- **Inductive Hypothesis**: Assume $T(n/2) \leq c' \cdot \frac{n}{2} \log \frac{n}{2}$

$$T(n) \leq 2 \cdot \left(c' \cdot \frac{n}{2} \log \frac{n}{2}\right) + cn = c'n \log \frac{n}{2} + cn$$
$$= c'n(\log n - 1) + cn = c'n \log n - c'n + cn$$

- **Conclusion**: If we choose $c' \geq c$, then:

$$T(n) \leq c'n \log n - c'n + cn \leq c'n \log n \quad \rightarrow \quad T(n) = O(n \log n)$$

Done. ✅

Credits & Resources

Lecture materials adapted from:

- Stanford CS161 slides and lecture notes
  - https://stanford-cs161.github.io/winter2025/
- *Algorithms Illuminated* by Tim Roughgarden
  - https://algorithmsilluminated.com/

📕 Exercise (& Assignment #1 - Due: 3rd October, 23:59 KST)

It is known that when the group size is $m = 5$ in the Median of Medians algorithm, the `select` algorithm runs in linear time $O(n)$. In this assignment, you will **generalize the proof** to every **odd** group size $m \geq 5$. For simplicity, assume that the input array size n is a multiple of m. *(Just think about why this assumption makes the proof easier.)*

- Estimated time: 2 hours
- No late submission is accepted.
- This will be the first-and-last assignement before the midterm!

Hint 1. Bounding the Larger Partition

Let A_{greater} be the set of elements strictly greater than the pivot.

1. Find an upper bound function $f(n, m)$ such that

$$|A_{\text{greater}}| \;\leq\; f(n, m).$$

2. As a sanity check: when $m = 5$, we know

$$|A_{\text{greater}}| \;\leq\; \tfrac{7}{10}n.$$

Hint 2. Recurrence for Running Time

The running time satisfies the recurrence

$$T(n) \le T\big(\frac{n}{m}\big) + T\big(f(n, m)\big) + cn,$$

where:

- the first term is the recursive call to find the median of medians,
- the second term is the recursive call on the larger partition,
- the last term corresponds to linear-time work (such as partioning).

Hint 3. Prove the Linear Bound

Prove by induction that there exists a constant c' such that

$$T(n) \leq c'n \quad \text{for all } n$$

1. Show how to choose such c' in terms of c and m
   - For example, when $m = 5$, we can pick $c' = \max\{1, 10c\}$
   - In general, find a function g such that

     $$c' = \max\{1, g(c, m)\}$$

     guarantees the inductive step closes.
2. Conclude that $T(n) = O(n)$ for every **odd** $m \geq 5$.

What to Submit

> Both handwriting and typewriting are OK.

- A PDF file (≤ 2 pages) containing:
  - Your bound $f(n, m)$ and a brief justification.
  - A complete inductive proof with your explicit choice of c', i.e., $g(c, m)$.
  - A worked instance for $m = 7$ to illustrate your proof.
- Grading
  - Correct bound $f(n, m)$ with reasoning (40%)
  - Valid induction and explicit c' choice; i.e., correct $g(c, m)$ (30%)
  - Valid justification for such a constant c' always exists for every odd $m \geq 5$ (30%)
- Submission will be through the LMS (submission portal will open there).