Greedy Algorithms

Interval Scheduling:

Input: \( n \) intervals \((S_i, f_i)\)

Story: only 1 classroom.

Schedule as many classes as possible w/o conflicts.

If \((S_i, f_i)\) and \((S_j, f_j)\) do not overlap,

there is no conflict between them.

Output: the maximum number of intervals that we can schedule without conflict.

Example 1: \((1, 8), (2, 3), (4, 5), (6, 7)\).

\( \checkmark \) \( \checkmark \) \( \checkmark \)

\( \text{OPT} = 3 \).

Example 2: \((1, 5), (6, 10), (4, 7)\).

\( \checkmark \) \( \checkmark \) \( \checkmark \) \( \checkmark \)

\( \text{OPT} = 2 \).

Example 3: \( \checkmark \) \( \checkmark \) \( \checkmark \) \( \checkmark \)

\( \text{OPT} = 4 \)

Greedy: decide on a simple rule that selects the first interval and include it in the final solution.

\( \downarrow \)

Remove all intervals that conflict with the chosen interval.

1. Choose an interval that starts the earliest.

   This is not optimal (see Example 1).

2. Choose a shortest interval.

   This is not optimal (see Example 2).

3. Choose an interval with the fewest number of conflicts.

   This is not optimal (see Example 3).

4. Choose an interval that finishes the earliest.

   (break ties arbitrarily).
Correctness:
- We output a compatible set of intervals.
- We output an optimal solution.

Idea: compare the algorithm's solution with some optimal solution.

Proof: Fix an input.
Let $|A| = k$ be the set of intervals chosen by GREEDY.
Let $|O| = m > k$ be an optimal solution.
$A = (i_1, i_2, \ldots, i_k) \rightarrow$ Ordered from left to right.
$O = (j_1, j_2, \ldots, j_m) \rightarrow$

Claim: $\forall 1 \leq r \leq k. f(i_r) \leq f(j_r)$
Base case: $f(i_1) \leq f(j_1)$
Inductive step: Suppose $f(i_{r-1}) \leq f(j_{r-1})$.
Then $f(i_r) \leq f(j_r)$.

A: $\overbrace{\ldots \overbrace{i_{r-1}}^{i_r}}$ This impossible because GREEDY can pick the interval $j_r$.

O: $\overbrace{\ldots \overbrace{j_{r-1}}^{j_r}}$ Consequently, $f(i_k) \leq f(j_k)$. \Rightarrow Contradict that $|A| = k$. Because GREEDY can pick $j_k+1$.

Implementation and Runtime: (n intervals).

$\Theta(n \log n)$ Sort all intervals by finishing time (early to late).
(In the sorted list):
$\text{current} \_ \text{finish} = -\infty, \quad A = \emptyset.$
For $i = 1 \text{ to } n$
if ($\text{current} \_ \text{finish} \leq S_i$)
$\text{Add } i \text{ to } A; \quad \text{current} \_ \text{finish} = f_i;$
End if
End for
Return $A$.

Overall runtime $= \Theta(n \log n)$. 