# Greedy Algorithms

# Greedy Strategy

- Make a greedy choice at each step - something that locally optimises the solution by some measure

# Greedy Strategy

The toy problem

# Greedy Strategy

Basic Framework

- Select a greedy choice and admit to the solution
- Convert the problem into a sub-problem of smaller size
- Repeat

# Greedy Strategy

<u>To prove correctness</u>

1. Prove Feasibility

2. Prove optimality

- Exchange Argument

- Greedy stays ahead

# Greedy Strategy

Exchange Argument

Prove that there always exists a solution with the greedy choice as part of it.

- Consider any solution

- Transform it into the solution found by greedy algorithm without decreasing the quality

# Greedy Strategy

<u>Greedy stays ahead</u>

Measure the greedy algorithm's progress in a step-by-step fashion and prove that it does as good as any other algorithm

# Interval Scheduling

# Interval Scheduling

Input : I = $\{i_1, i_2, \ldots\}$, set of n intervals, $i_j = [f_j, l_j]$

Output: $I' \subseteq I$ such that for all $i_1, i_2 \in I', i_1 \cap i_2 = \emptyset$ and maximises $|I'|$.

# Interval Scheduling

Greedy choices

# Interval Scheduling

Greedy choices

# Interval Scheduling

$R = I, A = \varnothing$

while $R \neq \varnothing$

$i$ be the interval in R with the smallest finishing time

$A = A \cap \{i\}$

$I_i \subseteq R$ be the intervals that intersect $i$

$R = R \backslash I_i$

# Interval Scheduling

$R = I, A = \emptyset$

`while` $R \neq \emptyset$

$i$ be the interval in R with the smallest finishing time

$A = A \cap \{i\}$

$I_i \subseteq R$ be the intervals that intersect $i$

$R = R \backslash I_i$

Running Time ?

# Interval Scheduling

Proof of Correctness:

Feasibility : easy to see

Optimality :

# Interval Scheduling

A : solution returned by greedy algorithm

O : an optimal solution

To prove $|A| = |O|$

# Interval Scheduling

A : $\{a_1, a_2, \ldots a_k\}$

O : $\{o_1, o_2, \ldots, o_m\}$

Claim : For all $r \leq k, l_{a_r} \leq l_{o_r}$

# Interval Scheduling

Claim : For all $r \leq k, l_{a_r} \leq l_{o_r}$

By induction on r :

r=1 ; true by definition of the algorithm

Induction step:

let $l_{a_{r-1}} \leq l_{o_{r-1}}$

Assume, for contradiction, $l_{a_r} > l_{o_r}$

We know, $f_{o_r} > l_{a_{r-1}}$ ie., $o_r$ was in the input for A

A should have selected $o_r$ , a contradiction.

# Interval Scheduling

To prove $k \leq m$

# Interval Scheduling

To prove $k \leq m$,

Assume not.

We know that $l_{a_k} \leq l_{o_k}$

$o_{k+1}$ is still present in the input after the kth iteration.
A contradiction.

# Interval Covering

Input : A set of m intervals I, set of n points P on the real line such that every point is contained in at least one interval in I.

Output: A subset I′ of I, such that for every point $p \in P$, there exists $i \in I$ such that $p \in i$ and |I′| is minimised.