CS216: Program and Data Representation
University of Virginia Computer Science
Spring 2006 David Evans

# Lecture 7: Greedy Algorithms


http://www.cs.virginia.edu/cs216

## Menu

La vita e incerta - mangia il dolce per primo.

"Life is uncertain. Eat dessert first."
Ernestine Ulmer



## Greed is Good?

- Adam Smith, *An Inquiry into the Nature and Causes of the Wealth of Nations* (1776)
  "It is not from the benevolence of the butcher, the brewer, or the baker, that we expect our dinner, but from their regard to their own interests."
- Invisible hand: individuals acting on personal greed produces (nearly) globally optimal results



## Greedy Algorithms

- Make the locally best choice "myopically" at each step
  - Need to figure out what "dessert" is
- Hope it leads to a globally good solution
  - Sometimes, can prove it leads to an optimal solution
  - Other times (like phylogeny), non-optimal, but usually okay if you get lucky



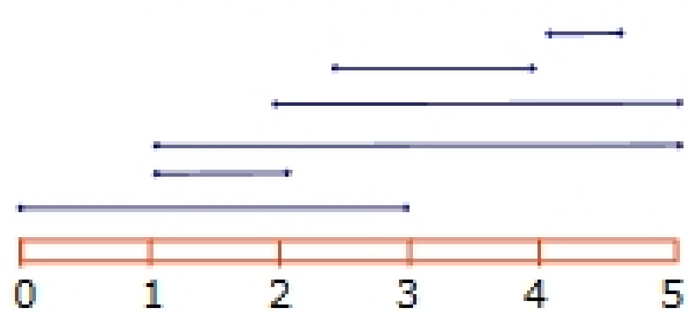
## Interval Scheduling Problem

- Input: $R$, a set of $n$ resource requests:
  $\{<s_0, f_0>, <s_1, f_1>, \ldots, <s_{n-1}, f_{n-1}>\}$
- Output: a subset $S$ of $R$ with no overlapping requests ($s_i > s_j < f_i$ for any $<s_i, f_i>, <s_j, f_j> \in S$) such that
  $|S| \geq |T|$ for any $T \subseteq R$ with no overlapping requests



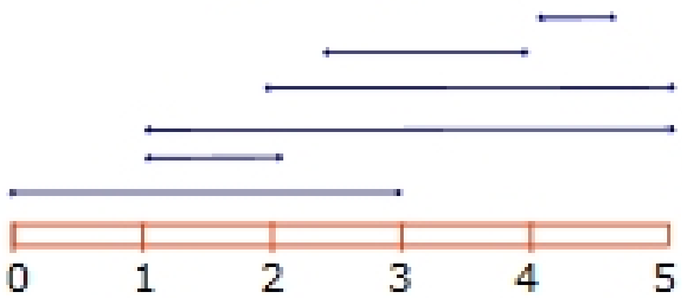
## Example

$R$ = {<0, 3>, <1, 2>, <1, 5>, <2, 5>, <2.5, 4>, <4, 4.5> }



## Solution

$R$ = {<0, 3>, <1, 2>, <1, 5>, <2, 5>, <2.5, 4>, <4, 4.5> }






## Brute Force Algorithm

- Try all possible subsets
  - Filter out ones with overlapping intervals
  - Pick the largest subset
- Running time
  - How many subsets? $2^n$
  - Constant work for each subset
  - $\Theta(2^n)$



## Greedy Approaches

- Need to pick best subset by making myopic decisions, one element at a time
- Many possible criteria for making myopic decision
  - Earliest starting time?
  - Latest ending time?
  - Shortest?



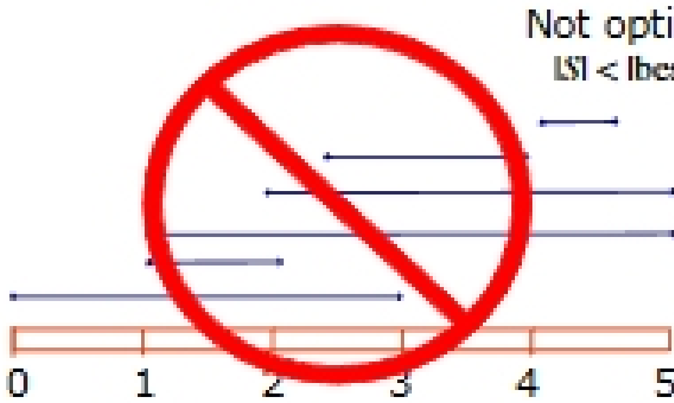
## Greedy Approach: Earliest Starting

Not optimal: $|S| < |best| = 3$




## Greedy Approach: Earliest Finishing

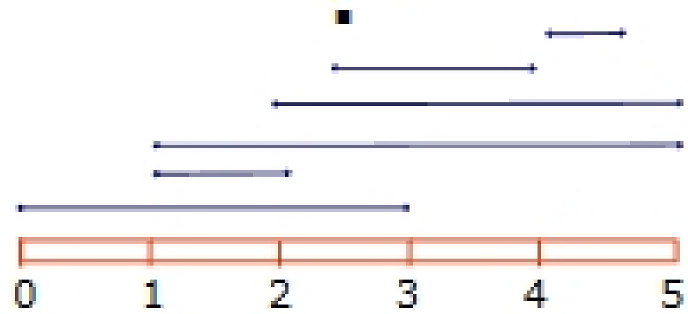
Not optimal: $|S| < |best| = 3$






## Greedy Approach: Shortest Length

?

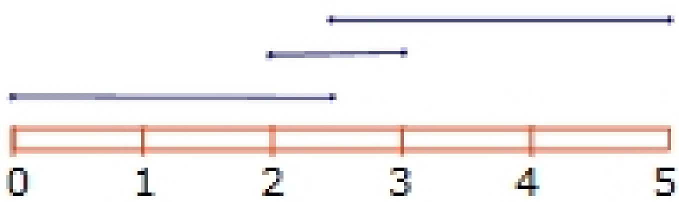


## Greedy Approach: Shortest Length

$R = \{<0, 2.5>, <2, 3>, <2.5, 5>\}$

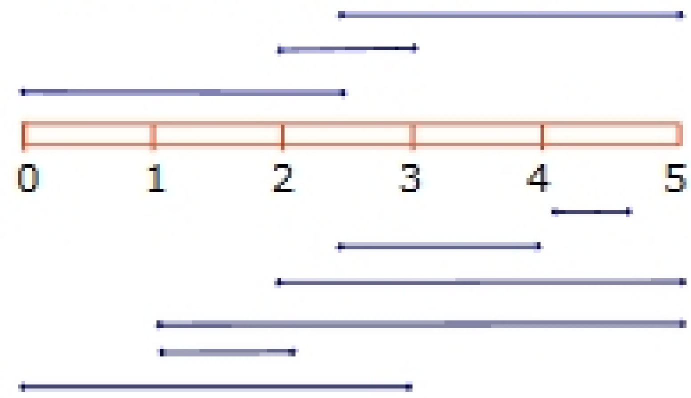
Not optimal:

$|S| < |best| = 2$






## Greedy Approach: Pick Earliest Finishing Time






## Greedy Algorithm: Running Time Analysis

- Straightforward implementation:
  - Search to find earliest finishing: $O(n)$
  - Eliminate matching elements: $O(n)$
  - Repeat (up to $n$ times): $O(n^2)$
- Smarter implementation:
  - Sort by finishing time: $O(n \log n)$
  - Go through list, selecting if non-overlapped: $O(n)$
  - Running time $\in O(n \log n)$



## Correctness?

- How to prove a greedy algorithm is ***non*optimal**
  - Find a counterexample: some input where the greedy algorithm does not find the best solution
- How to prove a greedy algorithm is **optimal**
  - By induction: always best up to some size
  - By exchange argument: swapping any element in solution cannot improve result



## Proof

- The greedy algorithm produces,

  $R = \{ r_0, \ldots, r_{k-1} \}$
- Suppose there is a better subset,

  $Q = \{ q_0, \ldots, q_{k-1}, q_k \}$
- Sort both by finishing time, so

  $f_{ri} < f_{rj}$ for all $0 \le i < j < k$

  $f_{qi} < f_{qj}$ for all $0 \le i < j < k+1$



## Proof

$R = \{ r_0, \ldots, r_{k-1} \}$

$Q = \{ q_0, \ldots, q_{k-1}, q_k \}$

$f_{ri} < f_{rj}$ for all $0 \le i < j < k$

$f_{qi} < f_{qj}$ for all $0 \le i < j < k+1$

Strategy:

1. Prove by induction $f_{r_i} \le f_{q_i}$ for all $i < k$
2. Then, since $f_{r_{k-1}} \le f_{q_{k-1}}$ if $q_k$ is valid, it would have also been added to $R$.