CS 312: Algorithms

Dan Sheldon

Mount Holyoke College

Last Compiled: October 1, 2018

Greedy Algorithms

We are moving on to our study of algorithm design techniques:

- Greedy
- Divide-and-conquer
- Dynamic programming
- Network flow

Let's jump right in, then characterize later what is means to be "greedy".

Interval Scheduling

- In the 80s, your only opportunity to watch a specific TV show was the time it was broadcast. Unfortunately, on a given night there might be multiple shows that you want to watch and some of the broadcast times overlap.

  Example on board

- You want to watch the highest number of shows. Which subset of shows do you pick?
- Fine print: assume you like all shows equally, you only have one TV, and you need to watch shows in their entirety.

Interval Scheduling

Let's formalize the problem

- Shows $1, 2, \ldots, n$ (more generally: "requests" to be fulfilled with a given resource)
- s_j: start time of show j
- f_j (sometimes $f(j)$): finish time of show j
- Shows i and j are compatible if they don't overlap.
- Set A of shows is compatible all pairs in A are compatible.
- Set A of shows is optimal. . . if it is compatible and no other compatible set is larger.

Greedy Algorithms

- Main idea in greedy algorithms is to make one choice at a time in a "greedy" fashion. (Choose the thing that looks best, never look back. . .)
- For shows, we will sort in some "natural order" and add shows to list one by one if they are compatible with the shows already chosen. Concretely:

$R \leftarrow$ be the set of all shows **sorted by some property**
$A \leftarrow \{\}$ ▷ selected shows
while R is not empty **do**
    Take first show i from R
    Add i to A
    Delete i and all overlapping shows from R
end while

Greedy Algorithm for Interval Scheduling

- What's a "natural order"?
  - *Start Time*: Consider shows in ascending order of s_j.
  - *Finish Time*: Consider shows in ascending order of f_j.
  - *Shortest Time*: Consider shows in ascending order of $f_j - s_j$.
  - *Fewest Conflicts*: Let c_j be number of shows which overlap with show j. Consider shows in ascending order of c_j.
- Sorting shows by finish time gives an optimal solution in examples. Let's try to prove that it will always be optimal.

Analysis

Let A be the set of shows returned by the algorithm when shows are sorted by finish time. What do we need to prove?

- A is compatible (obvious property of algorithm)
- A is optimal

We will prove A is optimal by a "greedy stays ahead" argument

Proof on board.

Ordering by Finish Time is Optimal: "Greedy Stays Ahead"

- Let $A = i_1, \ldots, i_k$ be the intervals selected by the greedy algorithm
- Let $O = j_1, \ldots, j_m$ be the intervals of some optimal solution O
- Assume both are sorted by finish time

```
A: |--i1--||---i2---| ... |---ik---|
O: |---j1---||---j2---|     ...      |----jm----|
```

- Could it be the case that $m > k$?
- Observation: $f(i_1) \leq f(j_1)$. The first show in A finishes no later than the first show in O.
- **Claim** ("greedy stays ahead"): $f(i_r) \leq f(j_r)$ for all $r = 1, 2, \ldots$. The rth show in A finishes no later than the rth show in O.

"Greedy Stays Ahead"

- **Claim**: $f(i_r) \leq f(j_r)$ for all $r = 1, 2, \ldots$
- **Proof** by induction on r
- **Base case** ($r = 1$): i_r is the first choice of the greedy algorithm, which has the earliest overall finish time, so $f(i_r) \leq f(j_r)$

Induction step:

- Assume inductively that $f(i_{r-1}) \leq f(j_{r-1})$
- Assume for sake of contradiction that $f(i_r) \geq f(j_r)$

```
A: |--i1--| ... |---i(r-1)---||-------ir------|
O: |---j1---| ... |---j(r-1)---||----jr-----|
```

- But it must be the case that j_r is compatible with the first $r-1$ shows in A, because (using induction hypothesis)

$$s(j_r) \geq f(j_{r-1}) \geq f(i_{r-1})$$

- Therefore, the greedy algorithm could have selected j_r instead of i_r. But j_r finishes sooner than i_r, which contradicts the algorithm.
- Therefore, it must be the case that $f(i_r) \leq f(j_r)$

Running Time?

$R \leftarrow$ be the set of all shows **sorted by some property**
$A \leftarrow \{\}$ ▷ selected shows
while R is not empty **do**
  Take first show i from R
  Add i to A
  Delete i and all overlapping shows from R
end while

$\Theta(n \log n)$ — dominated by sort

Running time analysis is usually easy for greedy algorithms

Algorithm Design—Greedy

Greedy: make a single "greedy" choice at a time, don't look back.

	Greedy
Formulate problem	?
Design algorithm	easy
Prove correctness	hard
Analyze running time	easy

Focus is on proof techniques. Next time: another proof technique.