

## CMPSCI 311: Introduction to Algorithms

Greedy: Exchange Arguments—Scheduling to Minimize Lateness

Dan Sheldon

University of Massachusetts

Last Compiled: February 16, 2017

## 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

- ▶ Last time: “greedy stays ahead” (inductive proof)
- ▶ This time: exchange argument

## Scheduling to Minimize Lateness

- ▶ You have a very busy month:  $n$  assignments are due, with different deadlines

Assignments:

1:  ---	(len=1, due=2)
2:  --- ---	(len=2, due=5)
3:  --- --- ---	(len=3, due=6)
4:  --- ---	(len=2, due=7)

Deadlines:

	d1		d2	d3	d4				
	---		---		---		---		---
0	1	2	3	4	5	6	7	8	9

- ▶ How should you schedule your time to “minimize lateness”?

## Scheduling to Minimize Lateness

Let’s formalize the problem. The input is:

- ▶  $t_j$  = length (in days) to complete assignment  $j$  (or “job”  $j$ )
- ▶  $d_j$  = deadline for assignment  $j$

What does a schedule look like?

- ▶  $s_j$  = start time for assignment  $j$  (selected by algorithm)
- ▶  $f_j = s_j + t_j$  finish time

How to evaluate a schedule?

- ▶ Lateness of assignment  $j$  is  $\ell_j = \begin{cases} 0 & \text{if } f_j \leq d_j \\ f_j - d_j & \text{if } f_j > d_j \end{cases}$
- ▶ Maximum lateness  $L = \max_j \ell_j$

**Goal:** find a schedule to make maximum lateness as small as possible

## Possible Greedy Approaches

- ▶ **Note:** it never hurts to schedule assignments consecutively with no “idle time”  $\Rightarrow$  schedule determined by order of assignments
- ▶ What order should we choose?
  - ▶ *Shortest Length:* ascending order of  $t_j$ .
  - ▶ *Earliest Deadline:* ascending order of  $d_j$ .
  - ▶ *Smallest Slack:* ascending order of  $d_j - t_j$ .
- ▶ Only *earliest deadline first* is optimal in all examples. Let’s prove it is always optimal.

## Exchange Argument (False Start)

Assume jobs ordered by deadline  $d_1 \leq d_2 \leq \dots \leq d_n$ , so the greedy ordering is simply

$$A = 1, 2, \dots, n$$

**Claim:**  $A$  is optimal

**Proof attempt:** Suppose for contradiction that  $A$  is not optimal. Then, there is an optimal solution  $O$  with  $O \neq A$

- ▶ Since  $O \neq A$ , there must be two jobs  $i$  and  $j$  that are out of order in  $O$  (e.g.  $O = 1, 3, 2, 4$ )
- ▶ Let’s swap  $i$  and  $j$  and show we get a *better* solution  $O'$
- ▶  $\Rightarrow O$  is *not* optimal. Contradiction, so  $A$  must be optimal.

**Problem?**  $O'$  may still be optimal. **Example.**

## Exchange Argument (Correct)

Suppose  $O$  optimal and  $O \neq A$ . Then we can modify  $O$  to get a new solution  $O'$  that is:

1. No worse than  $O$
2. Closer to  $A$  in some measurable way

$O(\text{optimal}) \rightarrow O'(\text{optimal}) \rightarrow O''(\text{optimal}) \rightarrow \dots \rightarrow A(\text{optimal})$

**High-level idea:** gradually transform  $O$  into  $A$  without hurting solution, thus preserving optimality.

**Concretely:** show 1 and 2 above.

## Exchange Argument for Scheduling to Minimize Lateness

Recall  $A = 1, 2, \dots, n$ . For  $S \neq A$ , say there is an **inversion** if  $i$  comes before  $j$  but  $j < i$ . **Claim:** if  $S$  has an inversion,  $S$  has a **consecutive inversion**—one where  $i$  comes immediately before  $j$ .

**Main result:** let  $O \neq A$  be an optimal schedule. Then  $O$  has a consecutive inversion  $i, j$ . We can swap  $i$  and  $j$  to get a new schedule  $O'$  such that:

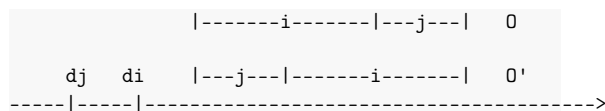
1. Maximum lateness of  $O'$  is no bigger than maximum lateness of  $O$
2.  $O'$  has one less inversion than  $O$

**Proof:**

1. On board / next slide
2. Obvious

## Proof of 1

Swapping a consecutive inversion ( $i$  precedes  $j$ ;  $d_j \leq d_i$ )



Consider the lateness  $\ell'_k$  of each job  $k$  in  $O'$ :

- ▶ If  $k \notin \{i, j\}$ , then lateness is unchanged:  $\ell'_k = \ell_k$
- ▶ Job  $j$  finishes earlier in  $O'$  than  $O$ :  $\ell'_j \leq \ell_j$
- ▶ Finish time of  $i$  in  $O' =$  finish time of  $j$  in  $O$ . Therefore

$$\ell'_i = f'_i - d_i = f_j - d_i \leq f_j - d_j = \ell_j$$

**Conclusion:**  $\max_k \ell'_k \leq \max_k \ell_k$ . Therefore  $O'$  is still optimal.

## Wrap-Up

For any optimal  $O \neq A$  we showed that we showed that we could transform  $O$  to  $O'$  such that:

1.  $O'$  is still optimal
2.  $O'$  has one less inversion than  $A$

$O(\text{optimal}) \rightarrow O'(\text{optimal}) \rightarrow O''(\text{optimal}) \rightarrow \dots \rightarrow A(\text{optimal})$

Since there are at most  $\binom{n}{2}$  inversions, by repeating the process a finite number of times we see that  $A$  is optimal.