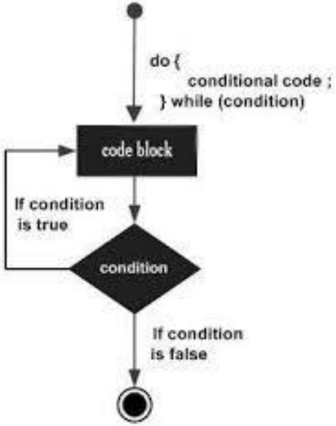
Lecture 11

Greedy Algorithms II



Announcements

- Homework 4 Reflections due Sunday night
- Group Meetings continue
- Quiz 1 due Saturday night
  - Up now
  - A few questions from each topic

Greedy Algorithms

We are moving on to our study of algorithm design

- **Greedy**
- Divide-and-conquer
- Dynamic Programming
- Network Flow

Last Time

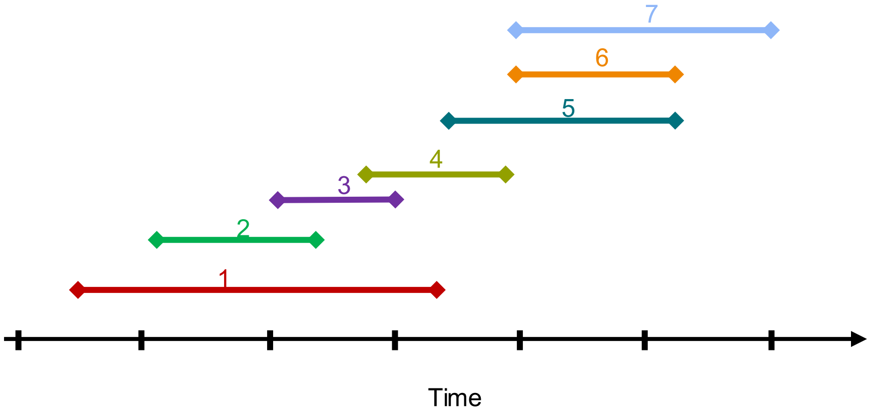
Introduced a new problem: interval scheduling

Arrived at an algorithm for scheduling

- ❖ Prioritize acts finishing first

Proof of Optimality Strategy

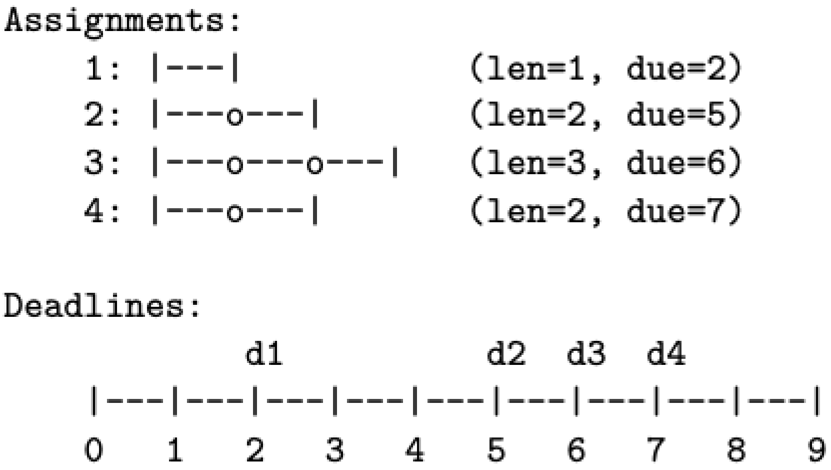
- ❖ "Greedy stays ahead"
- ❖ Pick an arbitrary optimal schedule, and show that the greedy schedule is at least as good as the optimal schedule at each step



Minimizing Lateness

New Problem

- n assignments are due, each with a different due date
- How should you schedule you time to minimize cumulative lateness?



```
Assignments:
    1: |---|                  (len=1, due=2)
    2: |---o---|              (len=2, due=5)
    3: |---o---o---|          (len=3, due=6)
    4: |---o---|              (len=2, due=7)

Deadlines:
            d1          d2  d3  d4
    |---|---|---|---|---|---|---|---|---|
    0   1   2   3   4   5   6   7   8   9
```

Minimizing Lateness

Formalizing the problem

- t_j the length (in days) to complete assignment j
- d_j the deadline for assignment j

Inputs

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

Minimizing Lateness

Formalizing the problem

- t_j the length (in days) to complete assignment j
- d_j the deadline for assignment j

Inputs

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

How should we evaluate a schedule?

- Lateness of assignment j is $l_j = \begin{cases} 0 & \text{if } f_j \le d_j \\ f_j - d_j & \text{else} \end{cases}$
- Maximize lateness $L = \max_j l_j$

Minimizing Lateness

Formalizing the problem

- t_j the length (in days) to complete assignment j
- d_j the deadline for assignment j

Inputs

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

How should we evaluate a schedule?

- Lateness of assignment j is $l_j = \begin{cases} 0 & \text{if } f_j \le d_j \\ f_j - d_j & \text{else} \end{cases}$
- Maximize lateness $L = \max_j l_j$

Goal: schedule so maximum lateness is as small as possible

Exercise 1

True or false: an algorithm to minimize maximum lateness will also find a schedule that is not late, if one exists.

a) True

b) False, because the lateness function L is not linear

c) False, because it minimizes the maximum lateness, whereas we want all jobs to zero lateness

Exercise 1

True or false: an algorithm to minimize maximum lateness will also find a schedule that is not late, if one exists.

a) True

b) False, because the lateness function L is not linear

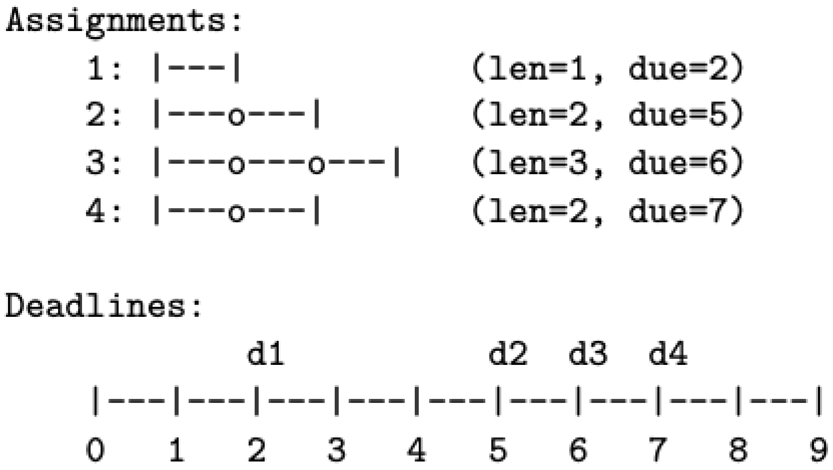
c) False, because it minimizes the maximum lateness, whereas we want all jobs to zero lateness

If the most expensive item in your shopping cart is less than \$5, then all items in your cart are less than \$5

Possible Greedy Approaches

A good schedule will have no idle time

- Implies that schedule is determined by an ordering on the assignments



```
Assignments:
    1: |---|                (len=1, due=2)
    2: |---o---|            (len=2, due=5)
    3: |---o---o---|        (len=3, due=6)
    4: |---o---|            (len=2, due=7)

Deadlines:
           d1          d2  d3  d4
    |---|---|---|---|---|---|---|---|---|
    0   1   2   3   4   5   6   7   8   9
```

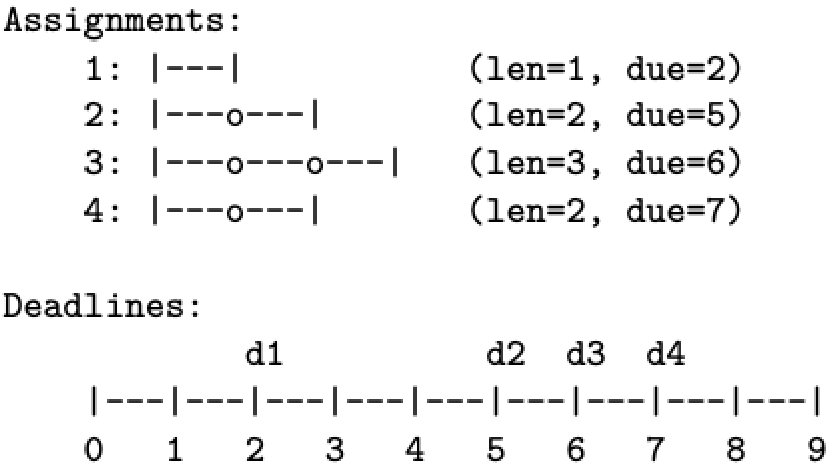
Possible Greedy Approaches

A good schedule will have no idle time

- Implies that schedule is determined by an ordering on the assignments

Possible Orderings:

- Shortest Length: ascending order of t_j
- Smallest Slack: ascending order of $d_j - t_j$
- Earliest Deadline: ascending order of d_j



```
Assignments:
    1: |---|                (len=1, due=2)
    2: |---o---|            (len=2, due=5)
    3: |---o---o---|        (len=3, due=6)
    4: |---o---|            (len=2, due=7)

Deadlines:
            d1          d2  d3  d4
    |---|---|---|---|---|---|---|---|---|
    0   1   2   3   4   5   6   7   8   9
```

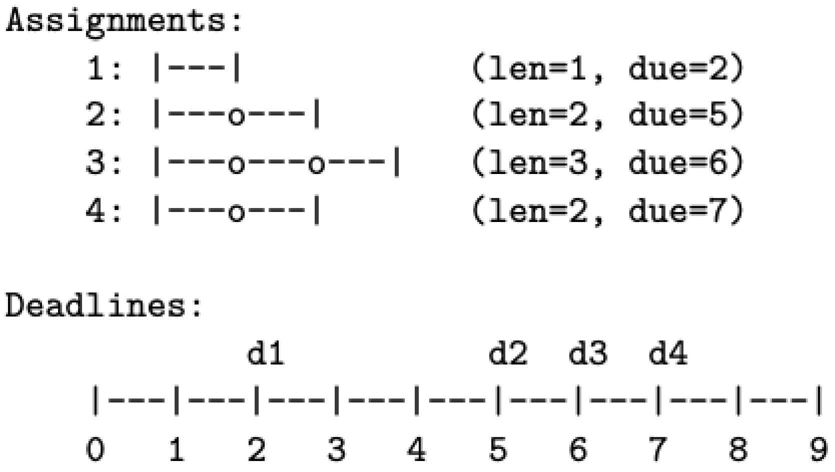
Possible Greedy Approaches

A good schedule will have no idle time

- Implies that schedule is determined by an ordering on the assignments

Possible Orderings:

- Shortest Length: ascending order of t_j
- Smallest Slack: ascending order of $d_j - t_j$
- Earliest Deadline: ascending order of d_j



```
Assignments:
    1: |---|              (len=1, due=2)
    2: |---o---|          (len=2, due=5)
    3: |---o---o---|      (len=3, due=6)
    4: |---o---|          (len=2, due=7)

Deadlines:
            d1          d2  d3  d4
    |---|---|---|---|---|---|---|---|---|
    0   1   2   3   4   5   6   7   8   9
```

Exercise 2

If two assignments have the same deadline, the "earliest deadline first" algorithm (EDF) should schedule:

a) The shortest job first

b) The longest job first

c) Does not matter

Exercise 2

If two assignments have the same deadline, the "earliest deadline first" algorithm (EDF) should schedule:

a) The shortest job first

b) The longest job first

c) Does not matter

Jointly, the two assignments require the same amount of time to complete

Identical Maximum Lateness

Claim: If in an EDF schedule, we swap two jobs with the same deadline, we get the same maximum lateness.

Identical Maximum Lateness

Claim: If in an EDF schedule, we swap two jobs with the same deadline, we get the same maximum lateness.

Proof:

- ❖ Since the schedules are EDF, all jobs with the same deadline are scheduled in a consecutive block.
- ❖ Among those, the last one has maximum lateness.
- ❖ That finishing time does not change by swapping schedules within the block.

Identical Maximum Lateness

Claim: If in an EDF schedule, we swap two jobs with the same deadline, we get the same maximum lateness.

Proof:

- ❖ Since the schedules are EDF, all jobs with the same deadline are scheduled in a consecutive block.
- ❖ Among those, the last one has maximum lateness.
- ❖ That finishing time does not change by swapping schedules within the block.

Corollary: All EDF schedules have the same maximum lateness.

Exchange Argument

Assume jobs are ordered by deadline $d_1 \leq d_2 \leq \cdots \leq d_n$, so that the greedy ordering is $A = 1,2, \ldots, n$

Claim: A is optimal with respect to maximum lateness.

Proof strategy:

- Gradually transform an arbitrary optimal solution O into A without hurting solution
- This preserves optimality with respect to maximum lateness
- Therefore, A is optimal

Exchange Argument Sketch

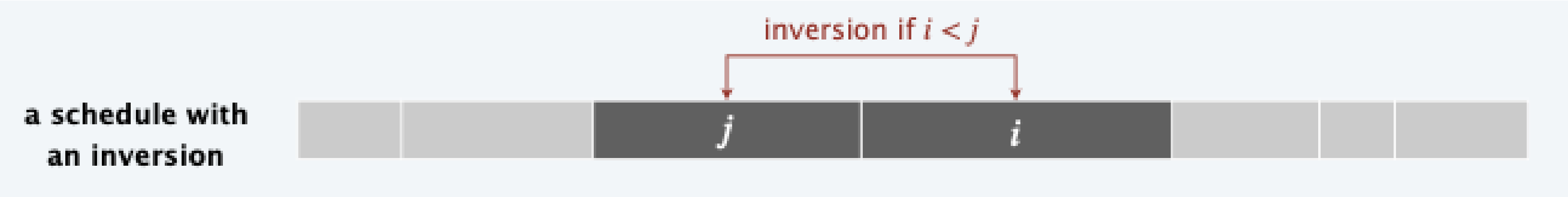
Suppose O is optimal and $O \neq A$. Then we can modify O to get a new solution O' that is

- No worse than O
- Closer to A in some measurable way

$$O(\text{optimal}) \to O'(\text{optimal}) \to O''(\text{optimal}) \to \ldots \to A(\text{optimal})$$

Exchange Argument

Recall $A = 1,2,\dots,n$. For $O \neq A$, we say there is an *inversion* if i comes before j but $j < i$ (thus $d_j \leq d_i$)

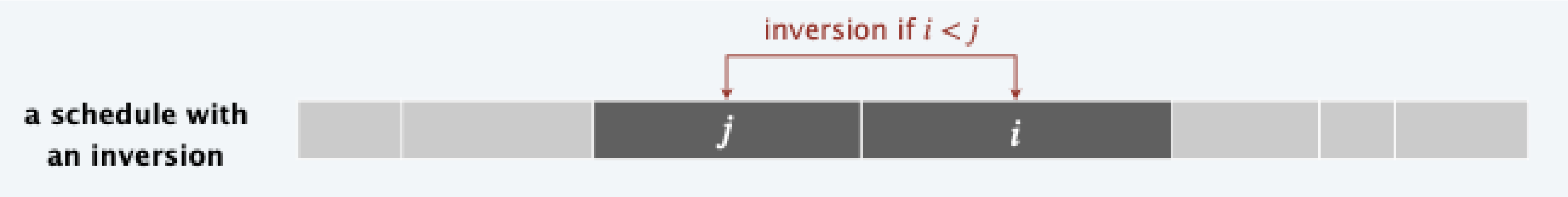


Claim: if O has an inversion, O has a *consecutive inversion*, where i comes immediately before j

- Why?

Exchange Argument

Recall $A = 1,2,\ldots,n$. For $O \neq A$, we say there is an *inversion* if i comes before j but $j < i$ (thus $d_j \leq d_i$)



Claim: if O has an inversion, O has a *consecutive inversion*, where i comes immediately before j

- Why?
- Let i, j be an inversion in O. Then $d_j \leq d_i$. Suppose some k is between i, j.
- If $d_k < d_j$ then j, k is a closer inversion; if $d_j \leq d_k$ then i, k is a closer inversion
- Repeat until inversion is adjacent

Exchange Argument

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 a get a new schedule O' such that

- O' has one less inversion than O
- Maximum lateness of O' is at most the maximum lateness of O

Exchange Argument

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 a get a new schedule O' such that

- O' has one less inversion than O
- Maximum lateness of O' is at most the maximum lateness of O

Proof:

- O' has one less inversion than O by construction
- Let l_i, l_i' be the lateness of i for O, O' respectively
- $l_k' \leq l_k$ for all $k \neq i, j$ and $l_i' \leq l_i$ (since moving i backward can only decrease lateness for i)

Exchange Argument

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 a get a new schedule O' such that

- O' has one less inversion than O
- Maximum lateness of O' is at most the maximum lateness of O

Proof:

- O' has one less inversion than O by construction
- Let l_i, l_i' be the lateness of i for O, O' respectively
- $l_k' \leq l_k$ for all $k \neq i, j$ and $l_i' \leq l_i$ (since moving i backward can only decrease lateness for i)
- If job j is late, $l_j' = f_j' - d_j$

Exchange Argument

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 a get a new schedule O' such that

- ❖ O' has one less inversion than O
- ❖ Maximum lateness of O' is at most the maximum lateness of O

Proof:

- ❖ O' has one less inversion than O by construction
- ❖ Let l_i, l_i' be the lateness of i for O, O' respectively
- ❖ $l_k' \leq l_k$ for all $k \neq i, j$ and $l_i' \leq l_i$ (since moving i backward can only decrease lateness for i)
- ❖ If job j is late, $l_j' = f_j' - d_j = f_i - d_j$

Exchange Argument

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 a get a new schedule O' such that

- O' has one less inversion than O
- Maximum lateness of O' is at most the maximum lateness of O

Proof:

- O' has one less inversion than O by construction
- Let l_i, l_i' be the lateness of i for O, O' respectively
- $l_k' \leq l_k$ for all $k \neq i, j$ and $l_i' \leq l_i$ (since moving i backward can only decrease lateness for i)
- If job j is late, $l_j' = \underbrace{f_j' - d_j = f_i - d_j}_{j \text{ now finishes at time } f_i}$

Exchange Argument

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 a get a new schedule O' such that

- ❖ O' has one less inversion than O
- ❖ Maximum lateness of O' is at most the maximum lateness of O

Proof:

- ❖ O' has one less inversion than O by construction
- ❖ Let l_i, l_i' be the lateness of i for O, O' respectively
- ❖ $l_k' \leq l_k$ for all $k \neq i, j$ and $l_i' \leq l_i$ (since moving i backward can only decrease lateness for i)
- ❖ If job j is late, $l_j' = f_j' - d_j = f_i - d_j \leq f_i - d_i = l_i$

Exchange Argument

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 a get a new schedule O' such that

- O' has one less inversion than O
- Maximum lateness of O' is at most the maximum lateness of O

Proof:

- O' has one less inversion than O by construction
- Let l_i, l_i' be the lateness of i for O, O' respectively
- $l_k' \leq l_k$ for all $k \neq i, j$ and $l_i' \leq l_i$ (since moving i backward can only decrease lateness for i)
- If job j is late, $l_j' = f_j' - d_j = \underbrace{f_i - d_j \leq f_i - d_i}_{i < j \text{ implies } d_i \leq d_j} = l_i$

Exchange Argument

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 a get a new schedule O' such that

- O' has one less inversion than O
- Maximum lateness of O' is at most the maximum lateness of O

Proof:

- O' has one less inversion than O by construction
- Let l_i, l_i' be the lateness of i for O, O' respectively
- $l_k' \leq l_k$ for all $k \neq i, j$ and $l_i' \leq l_i$ (since moving i backward can only decrease lateness for i)
- If job j is late, $l_j' = f_j' - d_j = f_i - d_j \leq f_i - d_i = l_i$

Conclusion: $\max\limits_k l_k' \leq \max\limits_k l_k$. Therefore, O' is still optimal.

Exchange Argument

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 a get a new schedule O' such that

- ❖ O' has one less inversion than O
- ❖ Maximum lateness of O' is at most the maximum lateness of O

Proof:

- ❖ O' has one less inversion than O by construction
- ❖ Let l_i, l_i' be the lateness of i for O, O' respectively
- ❖ $l_k' \leq l_k$ for all $k \neq i, j$ and $l_i' \leq l_i$ (since moving i backward can only decrease lateness for i)
- ❖ If job j is late, $l_j' = f_j' - d_j = f_i - d_j \leq f_i - d_i = l_i$

Conclusion: $\max_k l_k' \leq \max_k l_k$. Therefore, O' is still optimal. We obtain A by repeating this process finitely many times; therefore, A is optimal.

Exercise 2

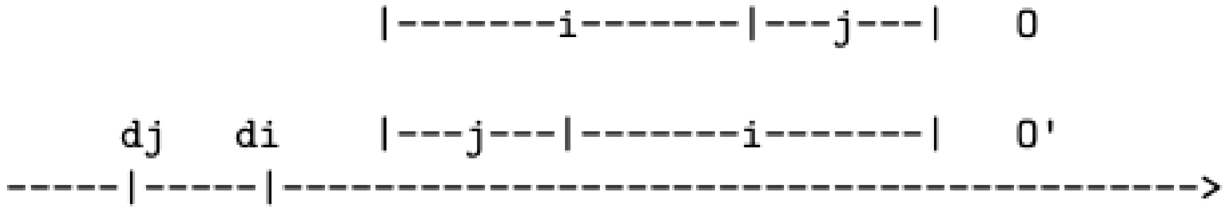
Consider the total lateness $l_i' + l_j'$ in the new schedule. Which fact about total lateness follows from our argument?

a) It is no more than $2l_i$

b) It is no more than $l_i + l_j$

c) It is no more than $2l_j$

d) None of the above



```
                 |-------i-------|---j---|   O

     dj    di    |---j---|-------i-------|   O'
-----|-----|---------------------------------->
```

Exercise 2

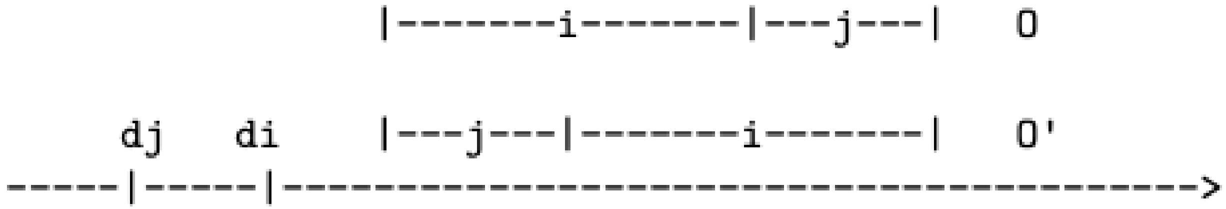
Consider the total lateness $l_i' + l_j'$ in the new schedule. Which fact about total lateness follows from our argument?

a) It is no more than $2l_i$

b) It is no more than $l_i + l_j$

c) It is no more than $2l_j$

d) None of the above



```
                |-------i-------|---j---|   O

     dj   di    |---j---|-------i-------|   O'
-----|-----|---------------------------------------->
```