CS 577- Intro to Algorithms

Greed

Dieter van Melkebeek

October 6, 2020

Outline

Outline

Discrete multivariate optimization

Outline

Discrete multivariate optimization

- System consisting of n components.
- Each component can be in any of a finite number of states.
- Want to set the states of the components so as to optimize an certain objective under certain constraints.

Outline

Discrete multivariate optimization

- System consisting of n components.
- Each component can be in any of a finite number of states.
- Want to set the states of the components so as to optimize an certain objective under certain constraints.

Paradigm

Outline

Discrete multivariate optimization

- System consisting of n components.
- Each component can be in any of a finite number of states.
- Want to set the states of the components so as to optimize an certain objective under certain constraints.

Paradigm

- Consider components in some order.

Outline

Discrete multivariate optimization

- System consisting of n components.
- Each component can be in any of a finite number of states.
- Want to set the states of the components so as to optimize an certain objective under certain constraints.

Paradigm

- Consider components in some order.
- Locally optimize setting of component based on prior components and settings only.

Outline

Discrete multivariate optimization

- System consisting of n components.
- Each component can be in any of a finite number of states.
- Want to set the states of the components so as to optimize an certain objective under certain constraints.

Paradigm

- Consider components in some order.
- Locally optimize setting of component based on prior components and settings only.

Correctness argument

Outline

Discrete multivariate optimization

- System consisting of n components.
- Each component can be in any of a finite number of states.
- Want to set the states of the components so as to optimize an certain objective under certain constraints.

Paradigm

- Consider components in some order.
- Locally optimize setting of component based on prior components and settings only.

Correctness argument

- Greed stays ahead

Outline

Discrete multivariate optimization

- System consisting of n components.
- Each component can be in any of a finite number of states.
- Want to set the states of the components so as to optimize an certain objective under certain constraints.

Paradigm

- Consider components in some order.
- Locally optimize setting of component based on prior components and settings only.

Correctness argument

- Greed stays ahead
- Exchanges

Interval Scheduling

Interval Scheduling

Problem

Interval Scheduling

Problem

Input: meetings $i \in [n]$ specified by start time $s_i \in \mathbb{R}$ and end time $e_i \in \mathbb{R}$.

Interval Scheduling

Problem

Input: meetings $i \in [n]$ specified by start time $s_i \in \mathbb{R}$ and end time $e_i \in \mathbb{R}$.

Ouput: $S \subseteq [n]$ such that no distinct intervals $[s_i, e_i)$ for $i \in S$ overlap and $|S|$ is maximized.

Interval Scheduling

Problem

Input: meetings $i \in [n]$ specified by start time $s_i \in \mathbb{R}$ and end time $e_i \in \mathbb{R}$.

Ouput: $S \subseteq [n]$ such that no distinct intervals $[s_i, e_i)$ for $i \in S$ overlap and $|S|$ is maximized.

Greedy algorithm

Interval Scheduling

Problem

Input: meetings $i \in [n]$ specified by start time $s_i \in \mathbb{R}$ and end time $e_i \in \mathbb{R}$.

Ouput: $S \subseteq [n]$ such that no distinct intervals $[s_i, e_i)$ for $i \in S$ overlap and $|S|$ is maximized.

Greedy algorithm

- Local criterion

Interval Scheduling

Problem

Input: meetings $i \in [n]$ specified by start time $s_i \in \mathbb{R}$ and end time $e_i \in \mathbb{R}$.

Ouput: $S \subseteq [n]$ such that no distinct intervals $[s_i, e_i)$ for $i \in S$ overlap and $|S|$ is maximized.

Greedy algorithm

- Local criterion
- Order

Natural Interval Orders

Natural Interval Orders

- Shortest first
- Fewest conflicts first
- Earliest start time first
- Earliest end time first
- Latest start time first
- Latest end time first

Algorithm

Algorithm

Powerpoint presentation

Algorithm

Powerpoint presentation

Complexity analysis

- $O(n \log n)$ time due to sorting.
- If meetings are given in sorted order: $O(n)$ time and $O(1)$ space for finding maximum value and producing schedule on-line.

Correctness – Greedy Stays Ahead

Correctness – Greedy Stays Ahead

Strategy

Design a quality measure for partial solutions such that:

Correctness – Greedy Stays Ahead

Strategy

Design a quality measure for partial solutions such that:

- For every valid solution S and every point in time k, the quality measure of the greedy solution G up to k is at least as good as S up to k.

Correctness – Greedy Stays Ahead

Strategy

Design a quality measure for partial solutions such that:

- For every valid solution S and every point in time k, the quality measure of the greedy solution G up to k is at least as good as S up to k.
- For a full solution, optimal quality measure implies optimal objective value.

Correctness – Greedy Stays Ahead

Strategy

Design a quality measure for partial solutions such that:

- For every valid solution S and every point in time k, the quality measure of the greedy solution G up to k is at least as good as S up to k.
- For a full solution, optimal quality measure implies optimal objective value.

Quality measures for earliest end time first

Correctness – Greedy Stays Ahead

Strategy

Design a quality measure for partial solutions such that:

- For every valid solution S and every point in time k, the quality measure of the greedy solution G up to k is at least as good as S up to k.
- For a full solution, optimal quality measure implies optimal objective value.

Quality measures for earliest end time first

Assume meetings numbered in greedy order.

Correctness – Greedy Stays Ahead

Strategy

Design a quality measure for partial solutions such that:

- For every valid solution S and every point in time k, the quality measure of the greedy solution G up to k is at least as good as S up to k.
- For a full solution, optimal quality measure implies optimal objective value.

Quality measures for earliest end time first

Assume meetings numbered in greedy order.

- cardinality $|S \cap [k]|$

Correctness – Greedy Stays Ahead

Strategy

Design a quality measure for partial solutions such that:

- For every valid solution S and every point in time k, the quality measure of the greedy solution G up to k is at least as good as S up to k.
- For a full solution, optimal quality measure implies optimal objective value.

Quality measures for earliest end time first

Assume meetings numbered in greedy order.

- cardinality $|S \cap [k]|$
- end time of the kth meeting in S

Cardinality as Quality Measure

Cardinality as Quality Measure

Claim

$(\forall k \in \mathbb{N})\, |G \cap [k]| \geq |S \cap [k]|$

Cardinality as Quality Measure

Claim
$(\forall k \in \mathbb{N})\, |G \cap [k]| \geq |S \cap [k]|$

Proof: Induction on k

Cardinality as Quality Measure

Claim

$(\forall k \in \mathbb{N})\,|G \cap [k]| \geq |S \cap [k]|$

Proof: Induction on k

- Base case: $k = 0$

Cardinality as Quality Measure

Claim

$(\forall k \in \mathbb{N})\, |G \cap [k]| \geq |S \cap [k]|$

Proof: Induction on k

- Base case: $k = 0$
- Inductive step $< k \rightarrow k$ for $k \notin S$

Cardinality as Quality Measure

Claim

$(\forall k \in \mathbb{N})\, |G \cap [k]| \geq |S \cap [k]|$

Proof: Induction on k

- Base case: $k = 0$
- Inductive step $< k \to k$ for $k \notin S$
- Inductive step $< k \to k$ for $k \in S$

Cardinality as Quality Measure

Claim

$(\forall k \in \mathbb{N})\, |G \cap [k]| \geq |S \cap [k]|$

Proof: Induction on k

- Base case: $k = 0$
- Inductive step $< k \to k$ for $k \notin S$
- Inductive step $< k \to k$ for $k \in S$
  Let ℓ be meeting in S right before k ($\ell = 0$ if there is none).

$$\begin{aligned} |G \cap [k]| &\geq 1 + |G \cap [\ell]| && \text{[greedy criterion]} \\ &\geq 1 + |S \cap [\ell]| && \text{[induction hypothesis]} \\ &= |S \cap [k]| && \text{[definition of } \ell\text{]} \end{aligned}$$

End Time as Quality Measure

End Time as Quality Measure

Definition

$$e_S(k) = \begin{cases} \text{end time of } k\text{th meeting in } S & \text{if it exists} \\ \infty & \text{otherwise for } k > 0 \\ -\infty & \text{for } k = 0 \end{cases}$$

End Time as Quality Measure

Definition

$$e_S(k) = \begin{cases} \text{end time of } k\text{th meeting in } S & \text{if it exists} \\ \infty & \text{otherwise for } k > 0 \\ -\infty & \text{for } k = 0 \end{cases}$$

Claim

$(\forall k \in \mathbb{N})\, e_G(k) \leq e_S(k)$

End Time as Quality Measure

Definition

$$e_S(k) = \begin{cases} \text{end time of } k\text{th meeting in } S & \text{if it exists} \\ \infty & \text{otherwise for } k > 0 \\ -\infty & \text{for } k = 0 \end{cases}$$

Claim

$(\forall k \in \mathbb{N})\, e_G(k) \leq e_S(k)$

Proof: Induction on k

End Time as Quality Measure

Definition

$$e_S(k) = \begin{cases} \text{end time of } k\text{th meeting in } S & \text{if it exists} \\ \infty & \text{otherwise for } k > 0 \\ -\infty & \text{for } k = 0 \end{cases}$$

Claim

$(\forall k \in \mathbb{N})\, e_G(k) \leq e_S(k)$

Proof: Induction on k

- Base case: $k = 0$

End Time as Quality Measure

Definition

$$e_S(k) = \begin{cases} \text{end time of } k\text{th meeting in } S & \text{if it exists} \\ \infty & \text{otherwise for } k > 0 \\ -\infty & \text{for } k = 0 \end{cases}$$

Claim

$(\forall k \in \mathbb{N})\, e_G(k) \leq e_S(k)$

Proof: Induction on k

- Base case: $k = 0$
- Inductive step $k - 1 \rightarrow k$

End Time as Quality Measure

Definition

$$e_S(k) = \begin{cases} \text{end time of } k\text{th meeting in } S & \text{if it exists} \\ \infty & \text{otherwise for } k > 0 \\ -\infty & \text{for } k = 0 \end{cases}$$

Claim

$(\forall k \in \mathbb{N})\, e_G(k) \leq e_S(k)$

Proof: Induction on k

- Base case: $k = 0$
- Inductive step $k - 1 \rightarrow k$

Corollary

$|G| \geq |S|$

From DP to Greed

From DP to Greed

- Consider meetings ordered earliest start time first.

From DP to Greed

- Consider meetings ordered earliest start time first.
- $\mathrm{OPT}(k) \doteq \mathrm{OPT}(\{k, k+1, \ldots, n\})$ for $1 \leq k \leq n+1$

From DP to Greed

- Consider meetings ordered earliest start time first.
- $\mathrm{OPT}(k) \doteq \mathrm{OPT}(\{k, k+1, \ldots, n\})$ for $1 \leq k \leq n+1$
- $\mathrm{OPT}(k) = \max(1 + \mathrm{OPT}(\mathrm{next}(k)), \mathrm{OPT}(k+1))$
  where $\mathrm{next}(k) \doteq \min\{\ell : k < \ell \leq n+1 \text{ and } s_\ell \geq e_k\}$

From DP to Greed

- Consider meetings ordered earliest start time first.
- $\mathrm{OPT}(k) \doteq \mathrm{OPT}(\{k, k+1, \ldots, n\})$ for $1 \leq k \leq n+1$
- $\mathrm{OPT}(k) = \max(1 + \mathrm{OPT}(\mathrm{next}(k)), \mathrm{OPT}(k+1))$
  where $\mathrm{next}(k) \doteq \min\{\ell : k < \ell \leq n+1 \text{ and } s_\ell \geq e_k\}$
- $\mathrm{OPT}(k) = \max(1 + \mathrm{OPT}(\mathrm{next}(k)),$

From DP to Greed

- Consider meetings ordered earliest start time first.
- $\mathrm{OPT}(k) \doteq \mathrm{OPT}(\{k, k+1, \ldots, n\})$ for $1 \leq k \leq n+1$
- $\mathrm{OPT}(k) = \max(1 + \mathrm{OPT}(\mathrm{next}(k)), \mathrm{OPT}(k+1))$
  where $\mathrm{next}(k) \doteq \min\{\ell : k < \ell \leq n+1 \text{ and } s_\ell \geq e_k\}$
- $$\mathrm{OPT}(k) = \max(1 + \mathrm{OPT}(\mathrm{next}(k)),$$
  $$1 + \mathrm{OPT}(\mathrm{next}(k+1)),$$

From DP to Greed

- Consider meetings ordered earliest start time first.
- $\mathrm{OPT}(k) \doteq \mathrm{OPT}(\{k, k+1, \ldots, n\})$ for $1 \leq k \leq n+1$
- $\mathrm{OPT}(k) = \max(1 + \mathrm{OPT}(\mathrm{next}(k)), \mathrm{OPT}(k+1))$
  where $\mathrm{next}(k) \doteq \min\{\ell : k < \ell \leq n+1 \text{ and } s_\ell \geq e_k\}$
- $$\begin{aligned}\mathrm{OPT}(k) = \max(&1 + \mathrm{OPT}(\mathrm{next}(k)),\\ &1 + \mathrm{OPT}(\mathrm{next}(k+1)),\\ &1 + \mathrm{OPT}(\mathrm{next}(k+2)),\end{aligned}$$

From DP to Greed

- Consider meetings ordered earliest start time first.
- $\mathrm{OPT}(k) \doteq \mathrm{OPT}(\{k, k+1, \ldots, n\})$ for $1 \leq k \leq n+1$
- $\mathrm{OPT}(k) = \max(1 + \mathrm{OPT}(\mathrm{next}(k)), \mathrm{OPT}(k+1))$
  where $\mathrm{next}(k) \doteq \min\{\ell : k < \ell \leq n+1 \text{ and } s_\ell \geq e_k\}$
- $$\begin{aligned}\mathrm{OPT}(k) = \max(&1 + \mathrm{OPT}(\mathrm{next}(k)),\\ &1 + \mathrm{OPT}(\mathrm{next}(k+1)),\\ &1 + \mathrm{OPT}(\mathrm{next}(k+2)),\\ &\ldots)\end{aligned}$$

From DP to Greed

- Consider meetings ordered earliest start time first.
- $\mathrm{OPT}(k) \doteq \mathrm{OPT}(\{k, k+1, \ldots, n\})$ for $1 \leq k \leq n+1$
- $\mathrm{OPT}(k) = \max(1 + \mathrm{OPT}(\mathrm{next}(k)), \mathrm{OPT}(k+1))$
  where $\mathrm{next}(k) \doteq \min\{\ell : k < \ell \leq n+1 \text{ and } s_\ell \geq e_k\}$
- $$\begin{aligned} \mathrm{OPT}(k) = \max(&1 + \mathrm{OPT}(\mathrm{next}(k)), \\ &1 + \mathrm{OPT}(\mathrm{next}(k+1)), \\ &1 + \mathrm{OPT}(\mathrm{next}(k+2)), \\ &\ldots) \\ = 1 + \mathrm{OPT}(&\mathrm{next}(k^*)) = \mathrm{OPT}(k^*) \\ &\text{where } k^* = \arg\min_{k \leq i \leq n} \mathrm{next}(i) \end{aligned}$$

From DP to Greed

- Consider meetings ordered earliest start time first.
- $\mathrm{OPT}(k) \doteq \mathrm{OPT}(\{k, k+1, \ldots, n\})$ for $1 \leq k \leq n+1$
- $\mathrm{OPT}(k) = \max(1 + \mathrm{OPT}(\mathrm{next}(k)), \mathrm{OPT}(k+1))$
  where $\mathrm{next}(k) \doteq \min\{\ell : k < \ell \leq n+1 \text{ and } s_\ell \geq e_k\}$
- $$\begin{aligned}\mathrm{OPT}(k) = \max(&1 + \mathrm{OPT}(\mathrm{next}(k)),\\ &1 + \mathrm{OPT}(\mathrm{next}(k+1)),\\ &1 + \mathrm{OPT}(\mathrm{next}(k+2)),\\ &\ldots)\\ = 1 + &\mathrm{OPT}(\mathrm{next}(k^*)) = \mathrm{OPT}(k^*)\\ &\text{where } k^* = \arg\min_{k \leq i \leq n} \mathrm{next}(i)\end{aligned}$$
- k^* is meeting with earliest deadline among $\{k, \ldots, n\}$.

From DP to Greed

- Consider meetings ordered earliest start time first.
- $\text{OPT}(k) \doteq \text{OPT}(\{k, k+1, \ldots, n\})$ for $1 \leq k \leq n+1$
- $\text{OPT}(k) = \max(1 + \text{OPT}(\text{next}(k)), \text{OPT}(k+1))$
  where $\text{next}(k) \doteq \min\{\ell : k < \ell \leq n+1 \text{ and } s_\ell \geq e_k\}$
- $$\begin{aligned}\text{OPT}(k) &= \max(1 + \text{OPT}(\text{next}(k)),\\ &\qquad 1 + \text{OPT}(\text{next}(k+1)),\\ &\qquad 1 + \text{OPT}(\text{next}(k+2)),\\ &\qquad \ldots)\\ &= 1 + \text{OPT}(\text{next}(k^*)) = \text{OPT}(k^*)\\ &\qquad \text{where } k^* = \arg\min_{k \leq i \leq n} \text{next}(i)\end{aligned}$$
- k^* is meeting with earliest deadline among $\{k, \ldots, n\}$.
- Attend k^*.

From DP to Greed

- Consider meetings ordered earliest start time first.
- $\mathrm{OPT}(k) \doteq \mathrm{OPT}(\{k, k+1, \ldots, n\})$ for $1 \leq k \leq n+1$
- $\mathrm{OPT}(k) = \max(1 + \mathrm{OPT}(\mathrm{next}(k)), \mathrm{OPT}(k+1))$
  where $\mathrm{next}(k) \doteq \min\{\ell : k < \ell \leq n+1 \text{ and } s_\ell \geq e_k\}$
- $$\begin{aligned} \mathrm{OPT}(k) = \max(&1 + \mathrm{OPT}(\mathrm{next}(k)), \\ &1 + \mathrm{OPT}(\mathrm{next}(k+1)), \\ &1 + \mathrm{OPT}(\mathrm{next}(k+2)), \\ &\ldots) \\ = 1 + &\mathrm{OPT}(\mathrm{next}(k^*)) = \mathrm{OPT}(k^*) \\ &\text{where } k^* = \arg\min_{k \leq i \leq n} \mathrm{next}(i) \end{aligned}$$
- k^* is meeting with earliest deadline among $\{k, \ldots, n\}$.
- Attend k^*.
- Continue process with $k \leftarrow \mathrm{next}(k^*)$.

DP vs Greed – moral

DP vs Greed – moral

Greedy algorithms never work!

Use dynamic programming instead!

What, never?
No, never!
What, *never*?
Well. . . hardly ever.[10]

DP vs Greed – moral

Greedy algorithms never work!

Use dynamic programming instead!

What, never?
No, never!
What, *never*?
Well. . . hardly ever.[10]

- Dynamic programming works in many settings. Greed only works in very simple settings.

DP vs Greed – moral

Greedy algorithms never work!

Use dynamic programming instead!

What, never?
No, never!
What, *never*?
Well. . . hardly ever.[10]

- Dynamic programming works in many settings. Greed only works in very simple settings.
- In very simple settings a greedy solution can sometimes be obtained by reasoning about a dynamic program.

DP vs Greed – moral

Greedy algorithms never work!

Use dynamic programming instead!

What, never?
No, never!
What, *never*?
Well. . . hardly ever.[10]

- Dynamic programming works in many settings. Greed only works in very simple settings.
- In very simple settings a greedy solution can sometimes be obtained by reasoning about a dynamic program.
- First develop a dynamic program. The consider whether it can be simplified into a greedy algorithm.

Knapsack Problem with Unit Values

Knapsack Problem with Unit Values

Problem

Knapsack Problem with Unit Values

Problem

Input: items $i \in [n]$ specified by weight $w_i \in \mathbb{Z}^+$
weight limit $W \in \mathbb{Z}^+$

Knapsack Problem with Unit Values

Problem

Input: items $i \in [n]$ specified by weight $w_i \in \mathbb{Z}^+$
weight limit $W \in \mathbb{Z}^+$

Ouput: $S \subseteq [n]$ such that
$\sum_{i \in S} w_i \leq W$ and $|S|$ is maximized.

Knapsack Problem with Unit Values

Problem

Input: items $i \in [n]$ specified by weight $w_i \in \mathbb{Z}^+$
weight limit $W \in \mathbb{Z}^+$

Ouput: $S \subseteq [n]$ such that
$\sum_{i \in S} w_i \leq W$ and $|S|$ is maximized.

Greedy algorithm

Knapsack Problem with Unit Values

Problem

Input: items $i \in [n]$ specified by weight $w_i \in \mathbb{Z}^+$
weight limit $W \in \mathbb{Z}^+$

Ouput: $S \subseteq [n]$ such that
$\sum_{i \in S} w_i \leq W$ and $|S|$ is maximized.

Greedy algorithm

- Local criterion

Knapsack Problem with Unit Values

Problem

Input: items $i \in [n]$ specified by weight $w_i \in \mathbb{Z}^+$
weight limit $W \in \mathbb{Z}^+$

Ouput: $S \subseteq [n]$ such that
$\sum_{i \in S} w_i \leq W$ and $|S|$ is maximized.

Greedy algorithm

- Local criterion
- Order