Algorithms in the Time of COVID19

Greedy algorithms[1] *- Recitation*[2] *5*

October 16, 2020

[1]Contact: Liangzu Peng (`lp2528@nyu.edu`).

[2]Hand-written notes available at: `https://www.dropbox.com/sh/x1z104c22d51pox/AACiJdDSKe2SDZw3qNljNApka?dl=0`

- Optimization: problems and algorithms
- Greedy algorithms
- Exercises

Optimization problems

Optimization problems:

$$\min_{x \in S} f(x) \tag{1}$$

- Does a solution to (1) exist?
- Is the solution to (1) unique?

Optimization algorithms

$$\min_{x \in S} f(x) \tag{2}$$

- Is there an (efficient) algorithm that finds a solution to (2)?

Proof of correctness:

- $x^{\dagger}$: the output produced by the algorithm.
- x^{*}: the (optimal) solution to (2).
- Prove that

$$f(x^{\dagger}) \leq f(x^{*})$$

or prove for any $x' \in S$ that

$$f(x^{\dagger}) \leq f(x').$$

Greedy algorithms

- Interval scheduling
- Total completion time
- Interval coloring
- Minimizing Lateness
- A famous greedy algorithm in signal processing and machine learning research: OMP[3]

[3] J. A. Tropp and A. C. Gilbert, "Signal Recovery From Random Measurements Via Orthogonal Matching Pursuit," in IEEE Transactions on Information Theory, vol. 53, no. 12, pp. 4655-4666, Dec. 2007, doi: 10.1109/TIT.2007.909108.

Greedy algorithms

$$\min_{x \in S} f(x) \tag{3}$$

Proof of correctness.
For any potential solution (x'), transform it into the solution $x^\dagger$ produced by the greedy algorithm via a series of steps.

- In step i we obtain some $x_i \in S$.
  - the transform path: $x' \to x_1 \to x_2 \to \cdots \to x^\dagger$.
- make sure that this path satisfies the property:

$$\mathscr{P} : f(x') \geq f(x_1) \geq f(x_2) \geq \cdots \geq f(x^\dagger). \tag{4}$$

Remark.
It often suffices to only consider step 1. (greedy choice property)

Greedy algorithms

Proof of correctness (cont.).
How can we creatively propose such steps?

- "Greedy Stays Ahead"[4]
- "Exchange Argument"

or use only one step which satisfies

- the greedy choice property,

after which the optimality of the greedy algorithm follows from

- the optimal substructure property.

[4] See for example `http://www.cs.cornell.edu/courses/cs482/2004su/handouts/greedy_ahead.pdf`

Interval scheduling

- Input: n intervals $\{[s_i, f_i]\}_{i=1}^n$
- Output: a maximal set of disjoint intervals
- Algorithm: repeatedly take the "compatible" interval with earliest finishing time

Interval scheduling: correctness proof

Let $f_1 \leq f_2 \leq \cdots \leq f_n$ (without loss of generality).
Observation.

- The output $(i_1, \ldots, i_k)$ of the greedy algorithm satisfies
  - $i_1 = 1$.
  - $i_1 < i_2 < \cdots < i_k$.
  - ...

Proof.
Let $j_1 < \cdots < j_m$ be a maximal set of disjoint intervals $(m \geq k)$.
Build a transform path.

- STEP 1: $(j_1, j_2, \ldots, j_m) \to (i_1, j_2 \ldots, j_m)$
- Quite naturally, can we:

$$(i_1, j_2, j_3 \ldots, j_m) \to (i_1, i_2, j_3 \ldots, j_m)?$$

  - intervals i_1 and i_2 are disjoint.
  - Risky: intervals i_2 and j_3 might have overlaps!

Interval coloring

- Input: n intervals $\{[s_i, f_i]\}_{i=1}^n$
- Goal: minimize the number of colors so that
  - each interval has one color.
  - overlapping intervals have different colors.

Exercises

Being Greedy?

Repeatedly choose maximal set of disjoint intervals?

Doesn't work

Color the earliest interval using available colors

Problem 1. Why does it not work for the interval coloring problem?

Minimize lateness

- Input: n jobs
  - job i has processing time $p_i > 0$ and deadline d_i
- Output: an ordering that minimizes the maximal lateness

$$\min \max_i L_i, \tag{5}$$

where $L_i := c_i - d_i$, with c_i the completion time of job i.

Exercises

Greedy Strategy

Finish the easiest thing first?

Doesn't work

Finish the one with earliest deadline

Problem 2. Why does it not work for the lateness minimization problem?

Problem 3. Given a finite set of points on the real line, determine the smallest set of unit-length closed intervals that contains all of the points.

A group of tourists are driving along a path with n touristic sites $1 \to 2 \to \cdots \to n$. Because of time constraints, only at most m sites can be visited ($m \leq n/2$). Site i has a value a_i which denotes the number of people in the group who would like to visit it, and we have $a_1 \geq a_2 \geq \cdots \geq a_n$. The tour guide wants to avoid short trips, so no two consecutive sites can be chosen. The group satisfaction from a subset $S \subset \{1, \ldots, n\}$ is $\sum_{j \in S} a_j$.

Problem 4. Devise a greedy algorithm that maximizes the group satisfaction and prove its correctness.

Exercises

Let $A = (a_1, \ldots, a_n)$ and $B = (b_1, \ldots, b_m)$ be two sorted lists of numbers, $n \leq m$. A pair (i, j) is called a *matching* if $a_i \leq b_j$. Two matchings (i, j) and (i', j') are considered distinct if and only if $i \neq i'$ and $j \neq j'$. For example, if $A = (1.1, 2.2, 3.3)$ and $B = (-2.2, 3.3, 4.4)$ then there are at most two distinct matchings, say $(1, 2)$ and $(2, 3)$, or say $(2, 2)$ and $(3, 3)$.

Problem 5. Given a non-negative integer $k \leq n$, for what kinds of A and B the maximum number of distinct matchings is exactly k? (give some examples of A and B.)

Problem 6. Design a greedy algorithm to find the maximum number of distinct matchings and prove its correctness.

Problem 7. Given $a_1 \geq \cdots \geq a_n$ and $b_1 \geq \cdots \geq b_n$, prove the rearrangement inequality

$$\sum_{i=1}^{n} a_i b_i \geq \sum_{i=1}^{n} a_i b_{\pi(i)} \geq \sum_{i=1}^{n} a_i b_{n-i+1}, \tag{6}$$

where $\pi : \{1, \ldots, n\} \to \{1, \ldots, n\}$ is a permutation.