PS#4 Greedy Algorithms

Çiçek Çavdar

April 19th 2010

Greedy Algorithms

- An algorithm is greedy if it builds a solution in small steps, choosing a decision at each step myoptically, not considering what may happen ahead to optimize some underlying criterion.
- For the optimality, therefore, we need to prove it “stays ahead” of all other solutions

Exercise #1

- Decide whether the following statement is T or F:
- Let G be an arbitrary connected, undirected graph with a distinct cost c(e) on every edge e. Suppose e* is the cheapest edge in G; that is c(e*)<c(e) for every edge e!=e*. Then there is a minimum spanning tree T of G that contains the edge e*.

Solution for Ex#1

- True, because
- e* is the first edge that would be considered by Kruskal's algorithm and so it would be included in the min. Spanning tree.

Exercise #2

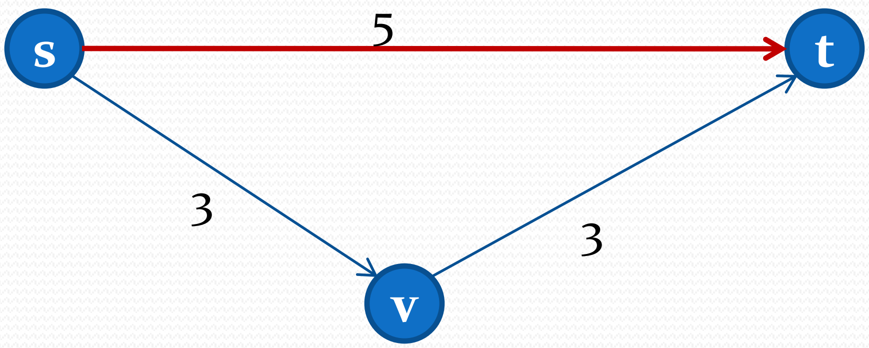
- Decide whether T or F:
  - (a) Given an instance of MST problem on G, with edge costs all positive and distinct. Let T be a MST for this instance. Suppose we replace each edge cost c_e by c_e^2. T is still a MST for this new instance?
  - (b) Given an instance of shortest s-t path problem on a directed G, with edge costs positive and distinct. Let P be a min cost s-t path for this instance. Suppose we replace each edge cost c_e by c_e^2. P is still a min cost s-t path?

Exercise 2.(a)

- If we feed the costs into Kruskal's algorithm, it will sort them in the same order, and hence put the same subset of edges in the MST.
- Note that: it is not enough just to say: "T because the edge costs have the same order after they are sorted." Because there are MST algos which just care about the relative order of the costs, not their actual values.

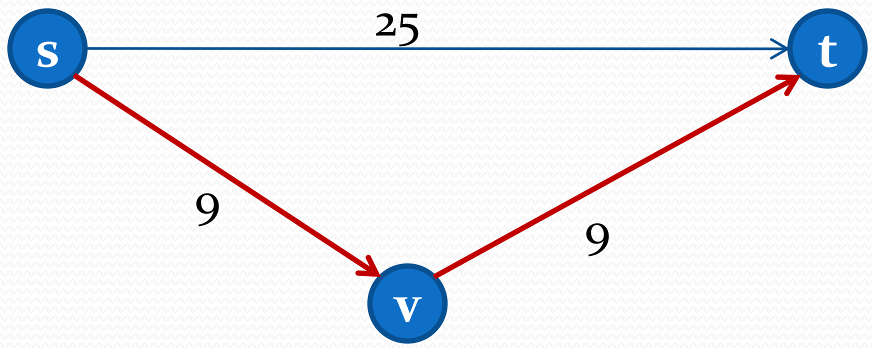
Exercise 2.(b)

- False



Exercise 2.(b)

- False



Exercise #3

- Shipping packages between N and B
- Each day there is at least 1 truck sent
- Each truck able to carry <=W
- Weight of each package i = w_i
- Only 1 truck in one station
- Boxes need to be shipped in the order they arrive.

Simple Greedy algo

- Pack the boxes in the order they arrive
- Whenever the next box doesn’t fit send the truck on its way.
- Can this greedy approch be improved?

Proof of optimality

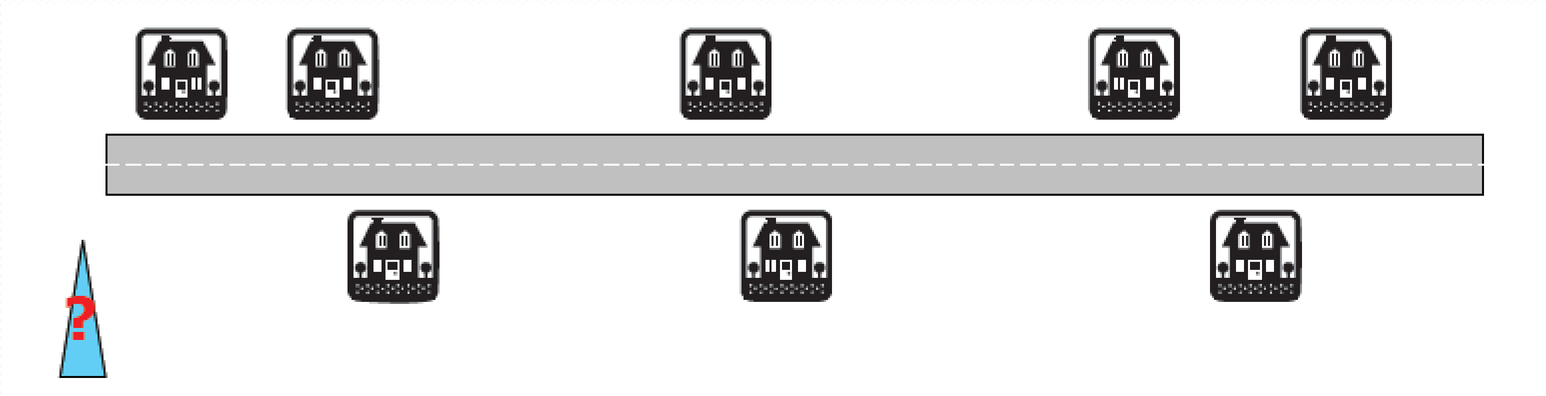
- Prove that greedy approach stays ahead of a possible optimum solution.
- We need to show that:
  - If greedy algo fits the first $b_1, b_2, \ldots b_j$ boxes into the first k trucks and another algorithm fits $b_1, b_2, \ldots b_i$ boxes in the first k trucks, then $i \le j$.
  - Let k be the number of trucks used by the greedy algorithm.

Proof by induction

- k=1 => it holds since the greedy algo fits as many boxes as possible in the first truck.
- Assume that it holds for k=n-1: The greedy algo fits j' boxes into the first n-1, and the other solution fits i'<=j'.
- k=n => the alternate sol. packs $b_{i'+1}$, $b_{i'+2}$,.. b_i. Thus since j'>=i' the greedy algo. is at least able to fit all the boxes $b_{j'+1}$, …,b_i into the n^{th} truck, and it can potentially fit more.

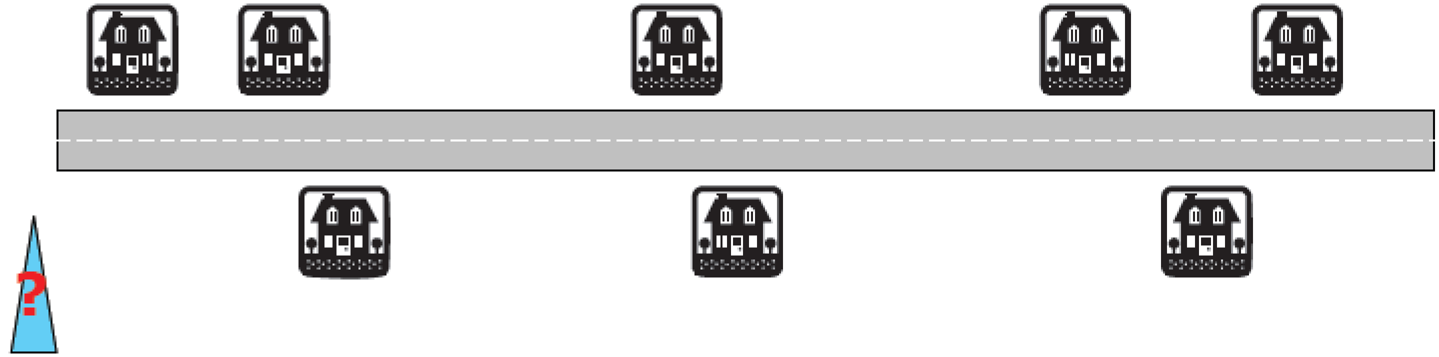
Exercise #5

- Base station placement: Place the min. # of base stations with the coverage of 4km for each house.



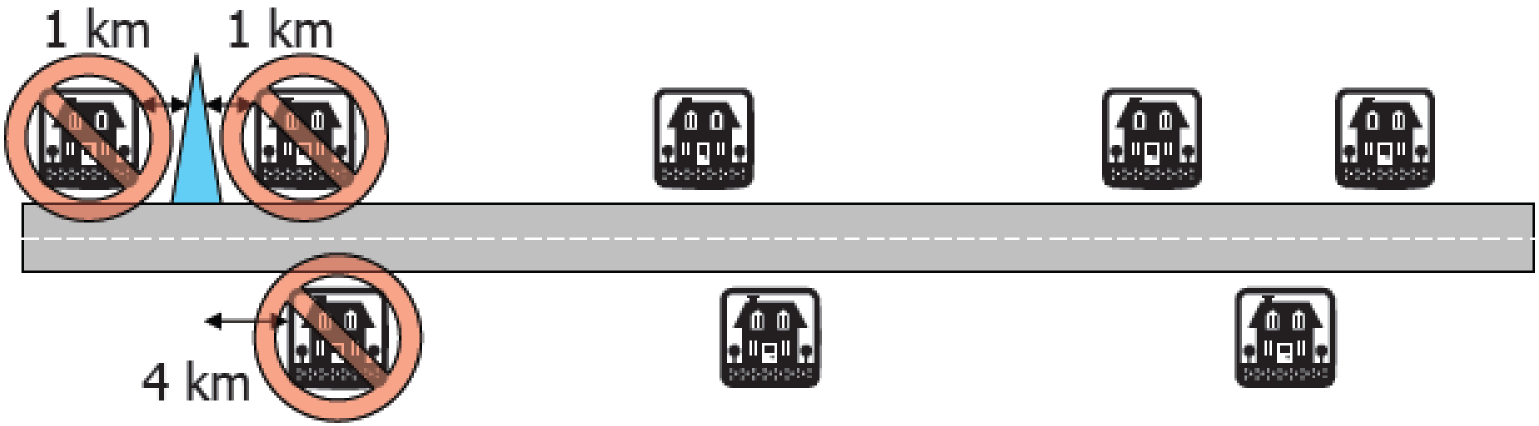
Ex. 5 Solution

- Define the position of the house from the west endpoint. Place the first base station at the easternmost position s_1, where the houses between o and s_1 are covered by this station.

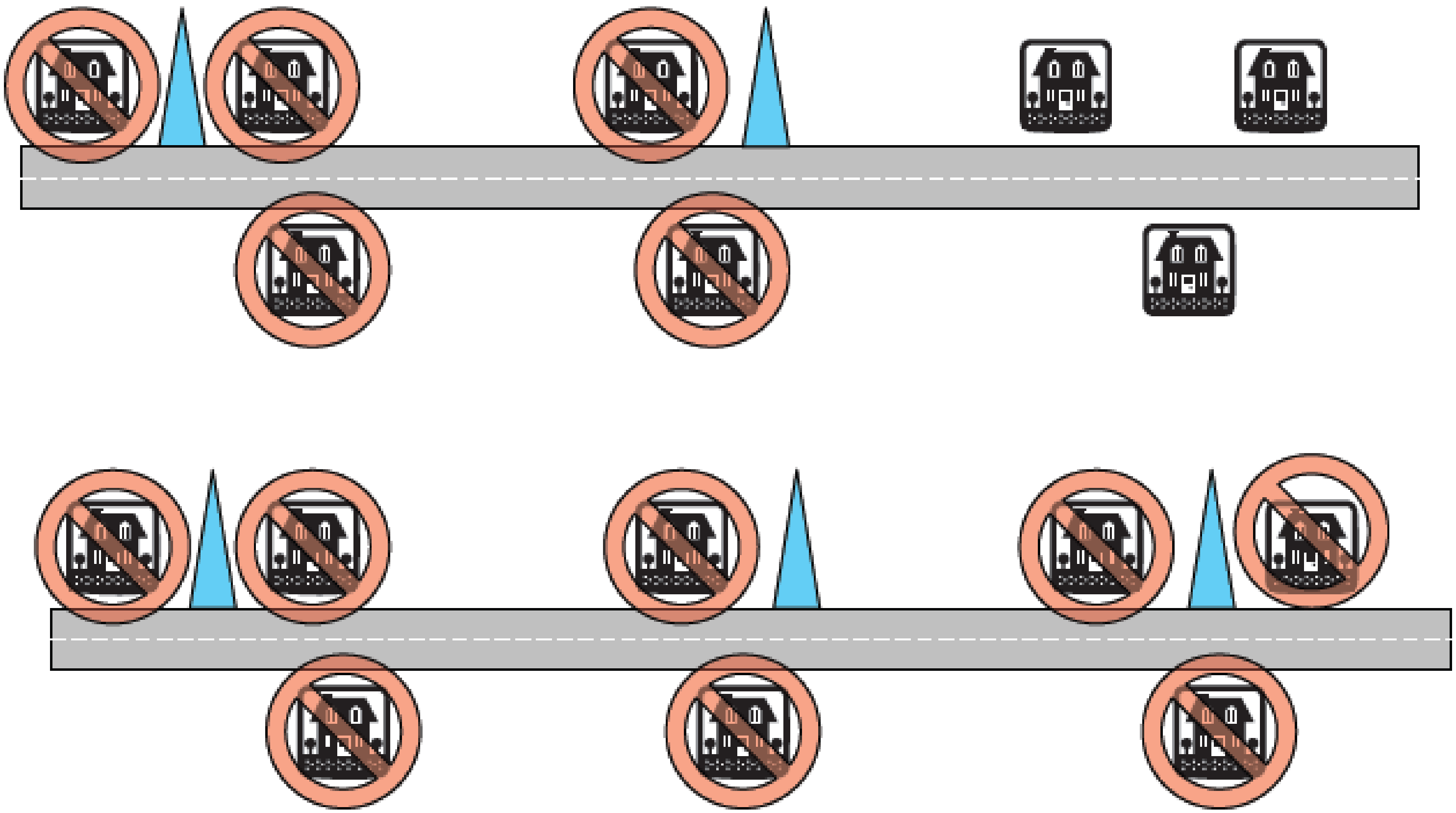


Simple Explanation

- Start at the western part of the road and move east until the first moment when there is a house 4 miles to the west
- Delete all the houses covered by this base station
- Continue as the first step



Move right until all are covered



Proof of optimality

- Generalize the problem: Having placed $\{s_1, s_2, .., s_i\}$, we place the new base station i+1 so that all the houses between s_i and s_{i+1} will be covered.
- Let $S=\{s_1, s_2, .., s_k\}$ denote the set of locations that the greedy algo has found.
- Let $T=\{t_1, t_2, .., t_m\}$ denote the set of locations that the optimal sol. has found. (sorted in increasing order)
- Show that k=m

Greedy “stays ahead” of optimal

- We claim for each i, $s_i >= t_i$.
- Proof by induction:
- i=1 true since we placed the station at the fartest east.
- Assume its true for some value for i>=1, this means the first S stations of S covers all the houses covered by T until i.
- Now if we add the i+1th station to S, (call it t_{i+1}), we will not leave any house between s_i and t_{i+1} uncovered.
- But we know that our algo chooses s_{i+1} as large as possible to cover all the houses between s_i and s_{i+1}. Then we should have $s_{i+1}>=t_{i+1}$.

Proof of k=m

- We already know that $k < m$ is false.
- if $(k>m)$ then $\{s_1,s_2,..,s_k\}$ fails to cover all houses.
- But $s_m>=t_m$, and so $\{t_1,t_2,..,t_m\}$ also fails to cover all houses,

  contradiction.
- Therefore k=m