CSCI 570 - Fall 2019 - HW 4

Junzhe Liu, 2270250947

September 24, 2019

1 Graded Problems

1.1 Suppose you were to drive from USC to Santa Monica along I-10. Your gas tank, when full, holds enough gas to go p miles, and you have a map that contains the information on the distances between gas stations along the route. Let $d_1 < d_2 < \ldots < d_n$ be the locations of all the gas stations along the route where d_i is the distance from USC to the gas station. We assume that the distance between neighboring gas stations is at most p miles. Your goal is to make as few gas stops as possible along the way. Give the most efficient algorithm to determine at which gas stations you should stop and prove that your strategy yields an optimal solution. Give the time complexity of your algorithm as a function of n.

In order to stop as possible, we shall keep on driving unless we cannot make it to the next gas stop.

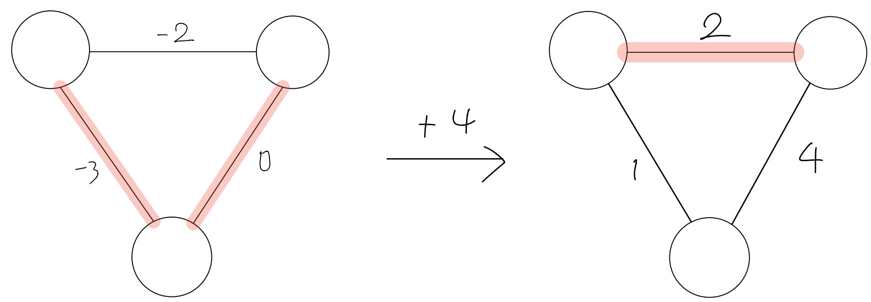
Algorithm 1 Minimizing number of gas stops

```
if d_n < p then We don't have to stop.
    return 0
end if
Initial d = 0
while haven't reach Santa Monica do
    scan all distances, choose the minimum i such that d + d_i < p and d + d_{i+1} > p.
    d ← d_i
    stop num +1
end while
```

This algorithm scan every distance at most once, therefore the complexity is $O(n)$.

1.2 Consider the following modification to Dijkstras algorithm for single source shortest paths to make it applicable to directed graphs with negative edge lengths. If the minimum edge length in the graph is $-w < 0$, then add $w + 1$ to each edge length thereby making all the edge lengths positive. Now apply Dijkstras algorithm starting from the source s and output the shortest paths to every other vertex. Does this modification work? Either prove that it correctly finds the shortest path starting from s to every vertex or give a counter example where it fails.

This modification apprently cannot work. A counterexample is shown in the following image:



Before adding $3 + 1 = 4$ to each edge, the minimum length from 1 to 2 is -3, but in the right graph, the minimum length turns to 2, the route changes too.

1.3 Kleinberg and Tardos, Chapter 4, Exercise 3.

Say n boxes arrive in the order $b_1, \dots, b_n$, and each box b_i has a positive weight w_i. To pack the boxes into N trucks preserving the order is to assign each box to one of the trucks $1, \dots, N$ so that:

- No truck is overloaded: the total weight of all boxes in each truck is less or equal to W.
- The order of arrivals is preserved: if the box b_i is sent before the box b_j, and b_i is assigned to truck x, b_j is assigned to truck y, and $x < y$, then it must be the case that b_i has arrived to the company earlier than b_j $(i < j)$.

We prove that the greedy algorithm uses the fewest possible trucks by showing that it ”stays ahead” of any other solution. Specifically, we consider any other solution and show the following. If the greedy algorithm fits boxes $b_1, b_2, \dots, b_j$ into the first k trucks, and the other solution fits $b_1, b_2, \dots, b_i$ into the first k trucks, then $i \leq j$. Note that this implies the optimality of the greedy algorithm, by setting k to be the number of trucks used by the greedy algorithm.

We will prove this claim by induction on k. The case $k = 1$ is clear; the greedy algorithm fits as many boxes as possible into the first truck. Now, assuming it holds for $k - 1$: the greedy algorithm fits $j^{'}$ boxes into the first $k - 1$, and the other solution fits $i^{' \leq j'}$. Now, for the k-th truck, the alternate solution packs in $b_{i'+1, \dots, b_i}$. Thus, since $i^{' \leq j'}$, the greedy algorithm is able at least to fit all the boxes $b_{j'+1, \dots, b_i}$ into the k-th truck, and it can potentially fit more. This completes the induction step, and hence the proof of optimality of the greedy algorithm.

1.4 Kleinberg and Tardos, Chapter 4, Exercise 5.

Imagine that the street is constructed like an array, and every house locates at a certain position indicted by the array index. Suppose there exist n houses, and their positions are sorted and numbered as: $d_1 < d_2 < \dots < d_n$. Each base station has an effective range of 4 miles to both sides, we can simplify this question as to use as few stations as possible to cover the interval $[d_1, d_n]$.

Algorithm 2 Cell phone base station cover range

```
Initialize d = d_1
while d_n is not covered: d_n < d + 8 do
    construct a base station at position d + 4.
    Find the smallest d_i which satisfies d_i > d + 8.
    d = d_i.
end while
```

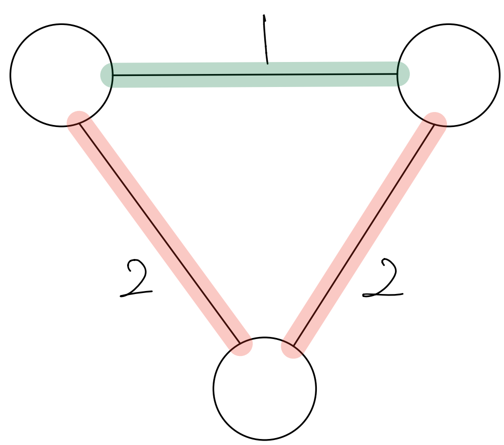
2 Practice Problems

2.1 Kleinberg and Tardos, Chapter 4, Exercise 4

2.2 Kleinberg and Tardos, Chapter 4, Exercise 8

Suppose there are two different MST T and $T^{'}$ in a graph. Since they are distinct, then there exist an edge $e^{'}$ in $T^{'}$ but is not included in T. If we add $e^{'}$ in T, we'll get a cycle C. Let e be the edge with the biggest edge cost in the cycle C, then according to cycle property, e should not belong to any MST in the graph. If $e = e^{'}$, then $T^{'}$ should not include $e^{'}$; if $e \neq e^{'}$, then T should include $e^{'}$ in its MST rather than e. Therefore $T = T^{'}$.

2.3 Kleinberg and Tardos, Chapter 4, Exercise 22



Not necessarily. Just like the image shows, suppose T is comprised by two red edges. The MST of this graph is a green edge with either two red edges. But T is not a MST.