CSCE 629

Jessica Salguero, UIN: 937005006 **Homework 2** February 7, 2025

Common Guidelines for submitting Homework & Project

- **No late homework/project submission will be accepted. Students should double check each submission to make sure correct files are submitted.**
- **By default, all work is solo; no collaboration allowed unless stated otherwise. Searching for solutions to homework problems from external resources (e.g., search online) is strictly forbidden.**
- Ensure clarity and conciseness in your work; avoid confusing and verbose explanations as they will not make any impact on your grade.
- Instead of using specific numerical examples, provide rigorous mathematical reasoning or a clear explanation demonstrating the correctness of your algorithm for all cases.
- In case of confusion, seek clarification from the TA or professor. DO NOT MAKE ASSUMPTIONS ABOUT EXPECTATIONS WHEN UNCERTAIN.
- You may use the provided LaTeX template to submit your work. Simply make a copy of this project on Overleaf, and if you encounter any problems using it, please contact the TA.

Problem Statement

Problem 1: Let us consider a long, straight country road with n houses scattered sparsely along it. (We can picture the road as a long line segment, with a western endpoint and an eastern endpoint, where the western point is at position 0, the eastern point is at position L ¿ 0, and the n houses are at positions x1, x2, · · · , xn, respectively, where 0 x1 ¡ x2 ¡ · · · ¡ xn L.) You want to place cell phone base stations at certain points along the road, so that every house is within (that is, less than or equal to) k miles of at least one of the base stations. Design an efficient algorithm that achieves this goal, using as few base stations as possible. Prove the correctness of your algorithm, and analyze its time complexity.

Main Idea

Describe here how you are planning to solve the problem.

For this scenario, everything would be occurring on a straight line and each additional house is closer to L than the previous one. This means that for each house, we need to decide if it is close enough to an existing cell phone base station to receive service or if a new one needs to be created. Since you need to make this decision at each house without immediately worrying about if the future houses have service, this calls for a greedy algorithm. This is

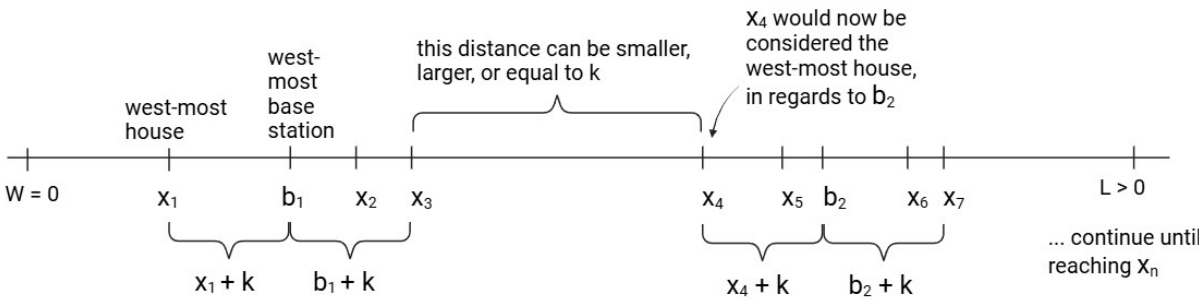

Figure 1: Problem breakdown

because once we select the location of a cell phone base station, that location is fixed; we then only need to worry about the remaining houses on the street, located east of that base.

To represent this scenario with a greedy algorithm:

- Pick the house closest to the west point (W = 0), this is the west-most house
- Create a base that is k distance away from that house, to the east
- Among the remaining houses, choose the ones that are within the correct range to get cell phone service from this existing base. This means that houses would need to be at a distance contained within the west-most house and the end of the current cell phone base's service range (k distance east from the base)
- Once there is a house that is not contained within the existing base's service range, then you need to create a new one. The 1st house that is too far east from the existing base would then become the west-most house for the new base.
- Repeat until you reach the last house on the street and reach the end of the road.

Pseudocode

Algorithm 1 Location of Cell Phone Base Stations

function BASE LOCATIONS(list)

Initialize global variables

- (current west-most house) will be initialized as equal to x1
- (current base) will be initialized as equal to x1 + k
- (current east-most position) will be initialized as equal to x1 + 2k. This is the farthest distance that the current base can service.
- (base locations) is a list, originally only containing the location of (current base). This is where you will store the location of each cell phone base station created.
- (L) is equal to eastern point of the road, where L is greater than 0

Loop through each of the individual positions (current house) in the sequence. The range will be from 2nd house (x2) to the last house (xn):

- Is (current house) contained between (current west-most house) and (current east-most position)?
  - Yes, it is less than or equal to (current east-most position):
    - This means that the (current house) is close enough to (current base) to receive service. Don't need to make any adjustments to variables. Continue to next loop
  - No, it is larger than (current east-most position). This means that the (current house) would not be close enough to be able to get service from an existing cell phone base, so you need to create another one:
    - Is the (current house) equal to (L)?
      - Yes, (current house) is equal to (L):
      - – Set (current west-most house) equal to (current house)
      - – Set (current base) equal to (current house)
      - – Add (current house) to the (base locations) list
      - – Set (current east-most position) equal to (current house)
      - No, (current house) is less than (L):
      - – Set (current west-most house) equal to (current house)
      - – Set (current base) equal to (current house + k). If (current house + k) is greater than L, set (current base) equal to (L)
      - – Add (current base) to the (base locations) list
      - – Set (current east-most position) equal to (current house + 2k). If (current house + 2k) is greater than (L), set (current east-most position) equal to (L)

After the loop is finished, return the list of base locations

end function

Proof of Correctness

Clearly articulate the key steps and reasoning behind your algorithm's correctness. Prove that your algorithm always finds a correct solution for all possible instances. ***Specific examples may be used to help explain your proof, but they cannot replace the rigorous proof.***

For this scenario (and other scenarios that require greedy algorithms), we can assume that there is one overall optimal solution for the problem as a whole. In order to reach this optimal solution, optimal choices must be made at each step along the way. This means that to reach your final optimal solution (S), S = S1 + S2 + S3 + ... + Sn for (n) choices you need to make. Your original problem/scenario can be broken down into smaller problems.

In theory, there are various ways that this large problem could be divided into smaller problems. However, since this specific scenario dealt with houses on a long straight road going from west to east, then it made sense to split up the problem by choosing the location of one base station at a time. Since we already knew the location of each individual house, you start with making that first house the farthest western location from your base station, to make that base station as far east along the road so it can service as many houses as possible. Then, when going in order of the houses along that long straight road, you continue to ask yourself if its within the correct distance to still get service from the first base station. Every house that is within service distance of the first base station ends up being the optimal solution for the location of base station 1 (S1). Then once you reach a house that isn't within service distance of base station 1, that becomes its own separate problem; the location of base station 1 (S1) is still optimal for those first couple of houses and now you only have to worry about the rest of the houses to the east. Base station 2 becomes its own separate problem that has an optimal solution; this continues until you reach the end of the road. Then, to reach your optimal solution for the entire base station problem, you add together the optimal solutions for each of the individual bases and the houses they provide cell phone service to.

Time Complexity

State and explain (if needed) the time complexity of your solution.

The time complexity of this algorithm is O(n). Since we can already assume that the list of house locations are already in order because each x value is larger than the previous one, where (x1) will be closest to the west point and (xn) will be closest to (L), then we only need to iterate through this list of houses once. For the creation of each base, you will ensure as many houses as possible are within k of the current base's locations. That means that the maximum number of loops will be equal to n, and time complexity will be O(n).

From the pseudocode:

- initializing global variables = O(1)

- The loop will range over n = O(n)
- Each round in the loop = O(1)

This means that overall, the time complexity is O(n)