CS 157: Assignment 1

Douglas R. Lanman
6 February 2006

Problem 4

Consider the following situation: power lines along a country road have been modified to carry broadband internet. Wi-Fi towers must be placed along the power line to provide the community with internet access. It is the goal of this write-up to develop efficient algorithms to place a minimum number of towers such that each house is sufficiently close to at least one tower.

In Part (a) the simple case of customers distributed on a line segment will be considered. In Part (b) the algorithm will be generalized such that customers can be distributed throughout a planar region rather than on a straight line segment. Finally, in Part (c), the algorithm will be further extended to more general power line paths. It will be shown that this problem is very similar to the interval scheduling task presented in class on 1/25/06.

Part (a): Customers located on a line segment

Given a line segment l, n locations of the customers on l, and a distance d, we wish to find a minimum number of points on l such that each location is at most d from some point. Give an efficient algorithm which returns a minimum size set of points.

High-level Description: This problem can be solved using a greedy algorithm. Consider a set of n points $\{x_i\}$ on the line l with $-\infty < x_i < \infty, \forall x_i$. Begin by sorting the points in ascending order such that $x_1 < x_2 < \cdots < x_n$. (Note that $x_i \neq x_j$ for $i \neq j$ since the locations of customers must be unique.) Next, place a tower at $x_1 + d$. Find the first point x_i to the right of $x_1 + d$ which is not included within the coverage region $\{x : x_1 - d \leq x \leq x_1 + d\}$ of the first tower. Place another tower at $x_i + d$ (i.e., as far to the right as allowed). Iterate this process until all points have been covered (always placing a tower as far to the right as possible). A pseudocode description of this algorithm is provided below in Find-Towers-Line.

```
Find-Towers-Line($x, d$)
1  sort $x$ into monotonically increasing order
2  $T \leftarrow \emptyset$
3  $c \leftarrow -\infty$
4  for $i \leftarrow 1$ to $length[x]$
5      do if $x[i] > c$
6            then $c \leftarrow x[i] + 2d$
7                 $T \leftarrow T \cup \{x[i] + d\}$
8  return $T$
```

Analysis of Running Time: The asymptotic worst-case running time of this algorithm is $O(n \log n)$. On line 1 of Find-Towers-Line the set of customer positions is sorted into ascending order. From class on 1/26/06 we know that sorting is $O(n \log n)$. For instance, we could use Merge-Sort from CLRS p. 32 to achieve this result [1]. Lines 2 and 3 can be evaluated in constant time (i.e., $O(1)$). Finally, the **for** loop on lines 4 through 7 must be evaluated for each customer, resulting in a linear running time (i.e., $O(n)$). Since $O(n \log n + n + 1) = O(n \log n)$ we have proved our claim. Note, if the input array x is sorted, then the running time is only O(n).

Proof of Correctness: First, note that the selected tower positions produced by Find-Towers-Line ensure that all customers are "covered" (i.e., are within a distance d of at least one tower). This is immediate from the placement of towers made on lines 4 through 7 – a tower is always placed such that any exposed point is covered and proceeds until all points have been considered.

Now we must prove that the proposed algorithm returns a set with minimum cardinality. We proceed by adopting the typical strategy for analyzing greedy algorithms: assume an ideal solution and seek a contradiction. Consider an optimal, lexicographically ordered set $T' = \{t'_1, t'_2, \ldots, t'_{m'}\}$ of m' towers that covers all customers with the minimum number of antennas. If the output of our algorithm is a set T with m towers, then assume $m > m'$ (i.e., that our solution is not optimal). Now compare the first optimal tower position t'_1 with the algorithm's first tower choice t_1. Since t_1 covers the first customer at x_1 and is as far as possible to the right of x_1, then $t'_1 \leq t_1$. As a result, we can substitute t_1 for t'_1 in T' without changing its cardinality (or causing a customer to go outside the coverage region). We can repeat this process for the i^{th} tower t'_i in T'. Since t_i is as far to the right as possible from the currently uncovered customer, we know that $t'_i \leq t_i$. As a result we can replace t'_i with t_i in T'. If we repeat this replacement policy we will eventually reach tower $t'_{m'}$ in T'. Since the modified set T' and T are identical up until tower m', then T' cannot be optimal because tower $t_{m'+1}$ is required to cover all neighbors by our algorithm. As a result, T' cannot be optimal and the set of towers T produced by Find-Towers-Line must have minimum cardinality.

Part (b): Customers located in a planar region

Given n locations of customers on a plane, a line segment l, and a distance d, we wish to find a minimum number of points on l such that each location is at most d from some point. We assume that all locations are at most a distance d from l. Give an efficient algorithm which returns a minimum size set of points.

Canonical Coordinates: We begin our analysis by introducing the notion of a canonical coordinate system. In this system, the power line is located along the $\hat{x}$-axis and customers are located within the region R given by $\{R : -\infty < x < \infty, -d \leq y \leq d\}$. This coordinate system is depicted in Figure 1 (customers are shown in blue).

Notice that, for any given customer at position (x_i, y_i), the minimum distance to the power line is given by y_i. Let us define the following interval $[s_i, f_i]$ with

$$s_i = x_i - \sqrt{d^2 - y_i^2} \tag{1}$$

$$f_i = x_i + \sqrt{d^2 - y_i^2} \tag{2}$$

as the "customer-required coverage interval". For a given customer, the Wi-Fi tower must be located within the region $s_i \leq x \leq f_i, y = 0$ in order to be at most a distance d from the user. (Notice that this interval corresponds to the intersection of the line l with a circle of radius d centered on the customer). From this specification it is clear that, when a customer is located a distance d from the power line, then $s_i = f_i = x_i$ and the tower can only be placed at $(x_i, 0)$ to cover this individual.

In general, we can parameterize an arbitrary line in the plane by a point (x_0, y_0) on the line and a counterclockwise angle θ (in radians) with respect to the $\hat{x}$-axis. We can transform this line and the set of customers $\{(x'_i, y'_i)\}$ to our canonical coordinate system $\{(x_i, y_i)\}$ by a simple translation and rotation using Equation 3.

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x'_i - x_0 \\ y'_i - y_0 \end{bmatrix} \tag{3}$$

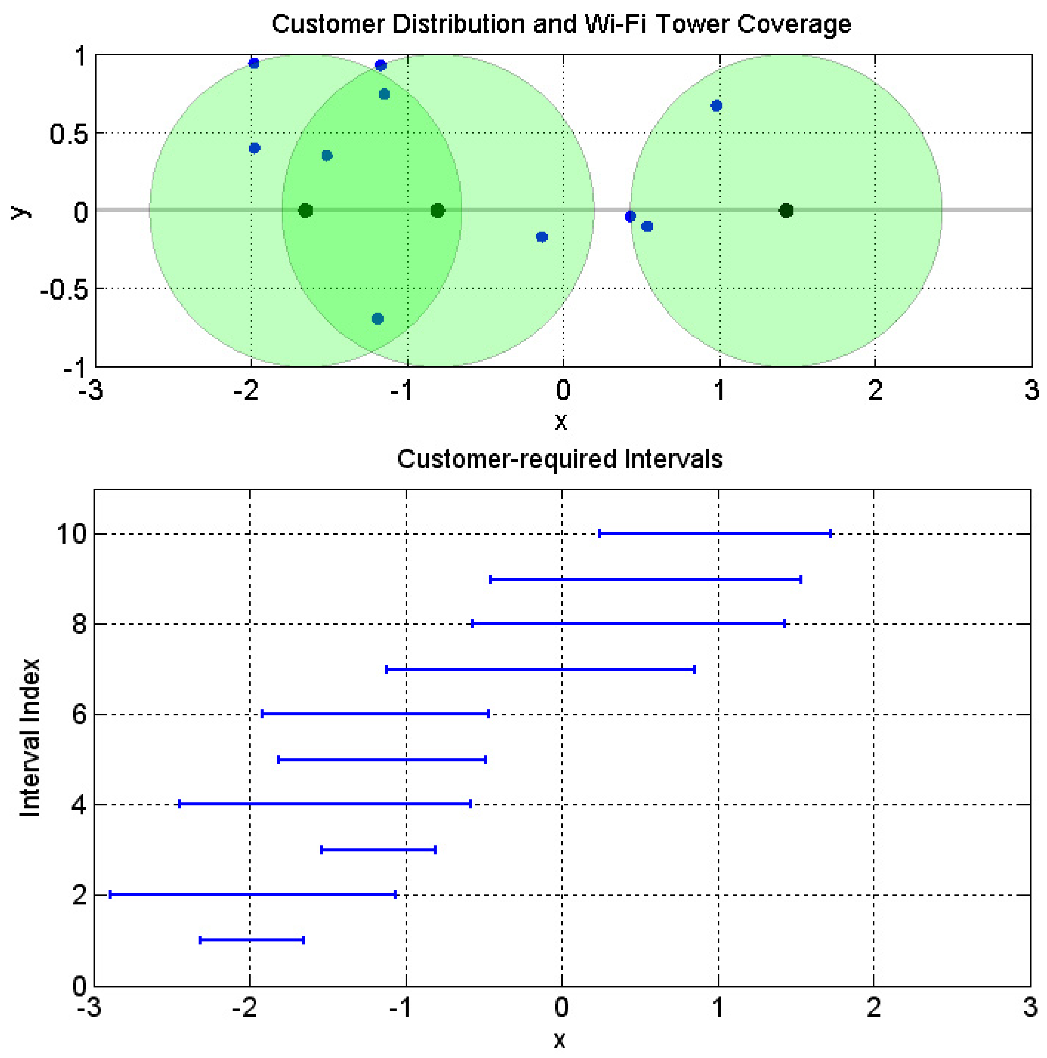


Figure 1: Sample Wi-Fi Tower placement achieved using FIND-TOWERS-PLANE.

Similarly, we can invert this transformation to convert points in the canonical system to our original system using Equation 4.

$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \tag{4}$$

Since Equations 3 and 4 can be used to transform any input line parameterized by $\{x_o, y_o, \theta\}$ to the canonical coordinate system shown in Figure 1, we can continue our analysis by only addressing the solution in the canonical frame.

High-level Description: In the canonical coordinate system we are given a set of points $\{(x_i, y_i)\}$. Using Equations 1 and 2 we can determine a set of intervals which constrain the location of the towers. As for the interval sorting task considered in class, we begin by sorting these intervals in order of monotonically increasing finishing time. (For example, examine the set of sorted intervals in Figure 1). Unlike Part (a), where each interval had length $2d$, the intervals can now have any length within $[0, 2d]$. Notice that, for the first interval in order of finishing time, we can once again place a tower as far to the right as possible at f_1. As in Part (a), we can exclude any customers that are covered by the first tower (i.e., those whose coverage intervals $[s_i, f_i]$ include f_1). Once we encounter a customer at x_i which isn't covered, we place a tower at f_i. This process continues until all customers have been considered. A pseudocode description of this algorithm is provided below in FIND-TOWERS-PLANE.

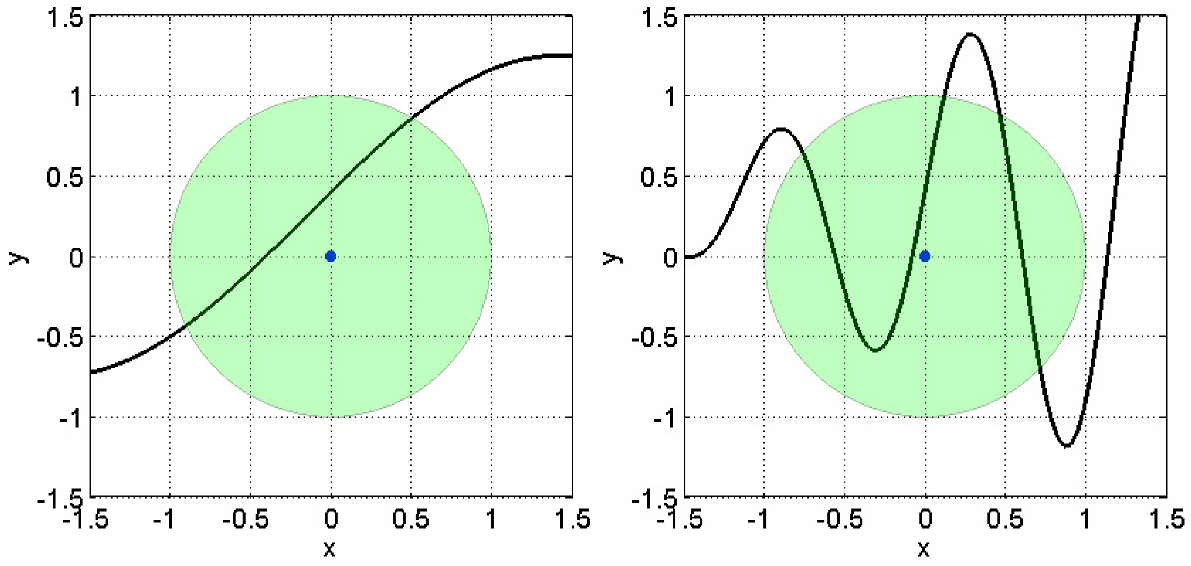
```
FIND-TOWERS-PLANE($x, y, d$)
 1  $n \leftarrow length[x]$
 2  transform each point $(x_i, y_i)$ to the canonical coordinate system
 3  for $i \leftarrow 1$ to $n$
 4      do $o \leftarrow \sqrt{d^2 - y[i]^2}$
 5         $s[i] \leftarrow x[i] - o$
 6         $f[i] \leftarrow x[i] + o$
 7  sort intervals $\{[s_i, f_i]\}$ into monotonically increasing finishing time $f_i$
 8  $T \leftarrow \emptyset$
 9  $c \leftarrow -\infty$
10  for $i \leftarrow 1$ to $n$
11      do if $s[i] > c$
12            then $c \leftarrow f[i]$
13                 $T \leftarrow T \cup \{c\}$
14  transform each tower's position to the input coordinate system
15  return $T$
```

Analysis of Running Time: The asymptotic worst-case running time of this algorithm is also $O(n \log n)$. As in Part (a), the sorting step on line 7 is $O(n \log n)$. The transformations on lines 2 and 14 can be implemented using Equations 3 and 4; since they must be applied for each point they have a running time of $O(n)$. Similarly, the determination of the coverage intervals on lines 3 through 6 will require looping over each customer and will having running time $O(n)$. As before, lines 10 through 13 require evaluating a condition at each customer and have a total running time of $O(n)$. The remaining lines can be evaluated in constant time and do not effect the asymptotic worst-case running time.

Proof of Correctness: First, note that the selected tower positions produced by FIND-TOWERS-PLANE ensure that all customers are covered. This is immediate from the placement of towers made on lines 10 through 13 – a tower is always placed such that any exposed point is covered and proceeds until all points have been considered.

In order to prove that the cardinality of T is minimal, we must first prove a subclaim; Note that, in Part (a), we made a claim that a tower could be replaced in T' without causing *any* customers to lose coverage. We begin by showing that, in general, if a tower t'_k located at (x'_k,0) provides coverage to a set of users at $\{(x_i, y_i)\}$, then we can replace this tower with one located at the minimum value of $\{f_i\}$ given by Equation 2. Put simply, if a tower provides coverage to a set of users, then we could move that tower to the right until any user was separated by a distance d. As shown in Figure 1, once the intervals $\{[s_i, f_i]\}$ have been sorted, we can determine if two users $\{i, j\}$ overlap if $s_j \leq f_i$ for $j > i$. As a result, if we move a tower to the right until it reaches the end of the earliest interval (by finishing time), all users originally covered by the tower will remain covered. In fact, by performing this operation we may actually include *more* users (see below).

Given this subclaim, the proof can proceed as in Part (a). Consider the first tower t_1 produced by FIND-TOWERS-PLANE. This tower must be located within $x \in [s_1, f_1]$. In general, the first tower t'_1 cannot be in the region $x > f_1$ (since the first user interval would not be covered). As a result, we have $t'_1 \leq t_1$. Given the previous subclaim, we can replace t'_1 with t_1 without causing *any* users covered by t'_1 to lose coverage. Now consider the i^{th} tower t'_i in T'. In general, this tower



(a) allowed path (b) prohibited path

Figure 2: Sample of power line paths allowed/prohibited by Find-Towers-General.

must satisfy $t_i' \leq t_i$, since t_i is placed as far to the right as possible in order to cover the currently uncovered user. As a result, we can once again invoke the subclaim and replace t_i' with t_i in T'. By induction, we find that all towers in T' can be replaced with those in T. As a result, T is also an optimal solution which produces a set of towers with minimum cardinality.

Part (c): Handling an arbitrary power line path

In your algorithm, how important is it that the line l is a line segment and not some arbitrary curve? If it is important, give an example which illustrates the problem. If it is not important, explain how you can adapt your algorithm.

High-level Description: While we previously restricted our analysis to line segments, Find-Towers-Plane can be extended to handle a broader class of planar curves. In Part (b) we found that each customer placed a restriction on the tower which provided its coverage (i.e., that it must be within the interval $[s_i, f_i]$). In general, if the power line curve Γ intersects every customer's coverage perimeter *no more than twice* then a modified form of Find-Towers-Plane can be used to determine a minimal set of tower locations.

Consider as an example the situation depicted in Figure 2(a). Here a customer is located at the origin and $d = 1$. The shaded green circle represents the locations where a tower could be placed in order to provide service to the customer – in general, this is a circle of radius d center at (x_i, y_i). The black curve represents the power line. Notice that, for this example, the power line enters and exits the the customer's coverage region exactly once. As a result, if the power line curve Γ is written in parametric form (with tracing parameter ξ), then the customer-required interval can be expressed as $s_i \leq \xi \leq f_i$, where s_i and f_i represent the value of ξ when the curve enters and leaves the region, respectively.

In general, if we can determine the set of intervals $\{[s_i, f_i]\}$ given by the intersection of the parametric curve Γ and each customer's coverage circle, then we can apply Find-Towers-General, shown below, to find the locations of the towers. This algorithm is very similar to that provided in Part (b), however the tower locations are now given in the tracing parameter ξ.

```
Find-Towers-General($x, y, d$)
1  determine intervals $\{[s_i, f_i]\}$ along parametric curve $\Gamma$
2  sort intervals $\{[s_i, f_i]\}$ into monotonically increasing finishing time $f_i$
3  $T \leftarrow \emptyset$
4  $\xi \leftarrow -\infty$
5  for $i \leftarrow 1$ to $length[x]$
6      do if $s[i] > \xi$
7            then $\xi \leftarrow f[i]$
8                 $T \leftarrow T \cup \{\xi\}$
9  return $T$
```

While handling more than simple line segments, Find-Towers-General can fail if the power line curve Γ intersects a customer's coverage circle more than twice. As an example, consider the case depicted in Figure 2(b). In this scenario, the power line enter and exists the coverage region twice. As a result, the interval $\{[s_i, f_i]\}$ no longer specifies all possible values of ξ, if s_i represents the first time Γ enters the circle and f_i represents the first exit. While a more general algorithm could be developed, it is not required for this write-up.

Analysis of Running Time: Assuming that the customer-required intervals $\{[s_i, f_i]\}$ can be evaluated with a running time no greater than $O(n \log n)$, then the asymptotic worst-case running time of this algorithm is also $O(n \log n)$. Once again, the overall running time of the algorithm is dependent on the running time of the sorting procedure.

Proof of Correctness: Follows directly from Part (b) with the assumption that each customer's coverage region intersects the power line curve Γ no more than twice.

References

[1] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Second Edition.* The MIT Press and McGraw-Hill, 2001.