Solution 1:

(a) For the basis case, suppose that \( S \) consists of a single vertex. We have \((v, e, f) = (1, 0, 1)\), which satisfies Euler’s formula. Otherwise, let \( S \) be any connected planar cell complex, consisting of at least one edge. If \( S \) has any vertex of degree 1 (that is, a vertex that is adjacent to only one edge), let \( S' \) be the cell complex resulting by removing this edge and this vertex. Clearly, we have \((v', e', f') = (v - 1, e - 1, f)\). Otherwise, every vertex of \( S \) has degree two or higher, and thus, \( S \) has at least one cycle. By the Jordan curve theorem, this cycle separates the plane into two parts, inside and outside. Thus, each edge has two different faces on each side of it. Remove any one edge on the cycle. This causes two faces to be merged into one face. We have \((v', e', f') = (v, e - 1, f - 1)\). In either case, \( S' \) is connected. By Euler’s formula \((v', e', f')\) satisfies Euler’s formula, and hence it is easy to see that so must \((v, e, f)\).

(b) Define the degree of a face to be the number of edges incident to the face. Let \( d \) denote the sum of degrees of all the faces. Since there are no multiple edges, each face has degree at least three, and hence \( d \geq 3f \). Also, by summing the degrees we count each edge twice (once for the face on either side), and hence \( d = 2e \). Thus, we have \( e \geq 3f/2 \). Plugging into Euler’s formula implies that \( 2 = v - e + f \leq v - (3f/2) + f = v - f/2 \), or equivalently \( f \leq 2(v - 2) \). We also have \( f \leq 2e/3 \). Plugging this into Euler’s formula yields \( 2 = v - e + f \leq v - e + (2e/3) = v - e/3 \). Therefore, \( e \leq 3(v - 2) \).

Solution 2:

(a) See Fig. 1.

(b) See Fig. 2 (bottom).

Figure 1: Solution to Problem 2(a).
Figure 2: Solution to Problem 2(b) and 2(c).

(c) See Fig. 2 (top).

(d) Observe that the subdivision in the unshared version differs from the trapezoidal map in the sense that each bullet path travels until it hits a segment that was inserted earlier in the insertion order. Consider a staggered stack of horizontal segments (see Fig. 3). Insert them in top to bottom order. When the $i^{th}$ segment is inserted, it will intersect both of the bullet paths of all the previous $i - 1$ segments. Thus, the total size is at least $\sum_{i=1}^{n} 2(i - 1) = \Omega(n^2)$.

Figure 3: Solution to Problem 2(d).

Solution 3:

(a) Computing the leftmost and rightmost points takes $O(n)$ time. Ignoring the times to bucket points, when a point is inserted in $O(1)$ time we can determine the edge lying immediately above or below the point. If we are below the edge, we ignore this point. If we are above, we start walking to the right and left until finding the left and right tangents. The time for this operation is $O(k + 1)$, where $k$ is the number of points deleted from the hull. The total time is $\sum_{i=1}^{n} (k_i + 1)$. Since each point can be deleted at most once, it follows that $\sum k_i \leq n$, and therefore the total sum is $O(n)$. The other steps of the algorithm (e.g., determining whether a vertex is a point of tangency and inserting a new vertex into a given position in the hull) take only $O(1)$ time each. Thus, the total time spent in adding and removing points is $O(n)$.

(b) Consider an arbitrary point $p$ of the set. What is the probability that this point will be rebucketed as a result of the $i^{th}$ insertion? We will use a backwards analysis. The key observation that we make use of
is that, after a point is added to the hull, the points to be rebucketed all lie one of the two vertical slabs passing through the left and right tangents that are incident to this point. Given \( p \), among the points that form the hull at time \( i \), let \( p_{j+1} \) denote the vertices on the edge whose vertical slab contains \( p \). By the prior observation, \( p \) was rebucketed at time \( i \) only if one of these two points was inserted last. Therefore, among the \( i \) points that have already been inserted, there are at most two, which if inserted last would cause \( p \) to be rebucketed. Thus it follows that the probability that \( p \) is rebucketed as a result of the \( i \) insertion (averaged over all possible insertion orders) is \( 2/i \). (It is important to note that all the points inserted so far should be counted among the possibilities, not just the points that lie on the convex hull.) Thus, the total expected number of times that \( p \) is rebucketed over the entire course of the algorithm is at most \( \sum_{i=1}^{n-2} 2/i = O(\log n) \). (We stop the sum at \( n-2 \), since two points were inserted before the randomization started.) Since this is true, independent of the point in question, it follows that the total expected rebucketing time over all the points is \( O(n \log n) \).

**Solution 4:** This is done by reduction to LP. For the sake of illustration, suppose that the \( x \)- and \( y \)-axes point to the right and up, respectively, and the \( z \)-axis is directed into the plane of this page. We can treat each rectangle as providing four constraints on the line, namely that it pass to the right of the left side, left of the right side, below the top side, and above the bottom side.

We parameterize each line in 3-space by a 4-tuple, \( (x_0, y_0, x_1, y_1) \), where \( (x_0, y_0) \) are the coordinates at which the line intersects the plane \( z = 0 \), and \( (x_1, y_1) \) are the coordinates at which the line intersects the plane \( z = 1 \) (see Fig. 3). The key observation is that the condition that a line passes to one side of some wall can be expressed as a linear inequality in this representation. In particular, given the 4-dimensional point \( p = (x_0, y_0, x_1, y_1) \), let \( q_0(p) = (x_0, y_0, 0) \) and \( q_1(p) = (x_1, y_1, 1) \). Let \( \ell(p) \) denote the line in 3-space that passes through \( q_0(p) \) and \( q_1(p) \). Any points on \( \ell(p) \) can be expressed as an affine combination of \( q_0 \) and \( q_1 \). In particular, we have

\[
\ell(p) = \{(1 - \alpha)q_0(p) + \alpha q_1(p) \mid \alpha \in \mathbb{R}\}.
\]

Consider the \( i \)th rectangle. Let \( (a_i, b_i) \) denote its lower-left \((x, y)\)-coordinates and \((c_i, d_i)\) denote its upper-right \((x, y)\)-coordinates, and let \( e_i \) denote its \( z \)-coordinate. The condition that \( \ell(p) \) satisfies the left and right side constraints is equivalent to saying that at \( z = e_i \), the \( x \)-coordinate of \( \ell(p) \) lies within the interval \([a_i, c_i]\), and the \( y \)-coordinate of \( \ell(p) \) lies within the interval \([b_i, d_i]\). The affine combination that leads to this point on \( \ell(p) \) is \( \alpha = e_i \). (This holds since the \( z \)-coordinate of \( q_0(p) \) and \( q_1(p) \) are 0 and 1, respectively, and hence the \( z \)-coordinate of \( (1 - e_i)q_0(p) + e_i q_1(p) = e_i \).) Thus, the \( x \)- and \( y \)-coordinates at this point are

\[
x = (1 - e_i)x_0 + e_i x_1 \quad \text{and} \quad y = (1 - e_i)y_0 + e_i y_1.
\]

In order to satisfy the constraint that the line stabs the rectangle, we have the two linear constraints:

\[
a_i \leq (1 - e_i)x_0 + e_i x_1 \leq c_i \quad \text{and} \quad b_i \leq (1 - e_i)y_0 + e_i y_1 \leq d_i.
\]
Thus, each rectangle gives rise to four linear constraints. Thus, we have four linear inequalities in four dimensional space. All that we are interested in establishing is that there exists a point \((x_0, y_0, x_1, y_1)\) that satisfies all these constraints. Thus, we may pick an arbitrary objective functions (e.g., \(c = (1, 1, 1, 1)\)), and then solve the associated LP. Given any point in the feasible region, the associated line is the resulting stabber.

Note that the inequalities in this case come in two classes. Those involving \(x_0\) and \(x_1\) and those involving \(y_0\) and \(y_1\). We could therefore decompose the problem into two separate LPs, each in 2-dimensional space.

**Solution 5:** We first observe that we can reduce this problem from one of moving a unit disk among a set of unit disks to that of moving a single point among a set of disks of radius 2. In particular, let \(p\) denote the center of the disk to be moved. Replace each other disk with a disk of radius 2. Observe that any point \(q\) in the plane that lies outside the union of the disks of radius 2 represents a valid placement of \(p\), and vice versa. This is because, a placement of \(p\) is valid if and only its center is at distance at least two from the nearest of the other disks of \(P\).

We could explicitly compute the union of these disks, but there is an easier way. We will first compute the Voronoi diagram of the points of \(P\). Intuitively, the Voronoi edges are the points that are locally farthest from any point of \(P\), and thus if a trajectory exists, except for its initial segment, the path will lie entirely on the edges of the Voronoi diagram (see Fig. 4(a)). To make this a bit more formal, we say that a valid path has maximal clearance if it is of the following form. Starting from an initial valid position, move directly away from the closest site until hitting a Voronoi edge. (If you never hit an edge, this path is already an escape route.) From this point, walk entirely along valid portions of Voronoi edges to an infinite edge. Clearly, such a path is valid. Conversely, suppose that there exists a valid escape path \(\pi\). For each point \(q \in \pi\) along this path (there are uncountably infinitely many of them), map it to a point \(q'\) by moving it directly away from its closest site to a Voronoi edge. Let \(\pi'\) be the resulting path. The point \(q'\) is not closer to its nearest site than \(q\) is. Therefore, if \(\pi\) is valid, then so is \(\pi'\). But \(\pi'\) is a maximal clearance path.

It suffices, therefore, to compute the valid portions of the Voronoi edges. Our approach will be to first trim away the portions of the Voronoi diagram that lie within distance 2 of some point of \(P\). Any path in the remaining trimmed Voronoi diagram, represents a valid trajectory for \(p\) (see Fig. 4(b)). To do this, consider each edge of the Voronoi diagram. Let \(p_i\) and \(p_j\) be the two sites defining this edge. Consider the disks of radius 2 centered at these sites. Remove the portion of the edge lying between them. In general, a middle section of each Voronoi edge may be removed as a result. In the remaining graph, consider the connected components that contain an edge that extends to infinity (see Fig. 4(a)). Such components reflect the escaping trajectories.

![Figure 5: Solution to Problem 5.](image-url)
at distance at least two from any other point of $P$, it follows that every point along this connecting segment is at distance at least two from any other point of $P$.

Computing the Voronoi diagram takes $O(n \log n)$ time. After that, in linear time we can traverse the graph, trim the edges, and compute the connected components that contain an edge to infinity. Given the start point $p$, we can determine whether any of its edges reside within such a component in $O(n)$ time. Thus, the overall running time is $O(n \log n)$.

**Solution to the Challenge Problem:**

The size of the final subdivision is proportional to the total number of segments stabbed by the vertical bullet paths. Observe that each bullet path travels until it hits a segment that was inserted earlier. (Clearly, it will stop when hitting such a segment at the time of its first insertion. Subsequent to this, whenever a newly inserted segment crosses the bullet path, it will not be trimmed, since such trimming will effectively merge two leaf nodes.)

Consider an endpoints of some segment $s \in S$. Consider each of the four bullet paths shot from the endpoints of $s$. We will show that the expected number of segments hit by this bullet path is $O(\log n)$. It will follow that the total complexity of the final diagram is $O(n \log n)$.

Let $s_1, s_2, \ldots, s_k$ denote the subset of all the segments that the vertical bullet path might intersect, sorted from closest to farthest from the endpoint (see Fig. 6). For $1 \leq i \leq k$, let $t_i$ denote the time at which segment $s_i$ is added and let $p_0$ be the time at which $s$ was added. The values of $t_i$s are random variables in the range from 1 to $n$, and because the insertion order is random, all permutations of the $t_i$'s are equally likely. Let $p_i$ be the probability that the bullet path cuts through segment $s_i$. In order for this to happen, $s$ must have been inserted before $s_1, s_2, \ldots, s_i$. That is, $t_0$ must be the smallest element of the set $\{t_0, t_1, \ldots, t_i\}$. Since all permutations of these values are equally likely, the probability of this event is $1/(i + 1)$. In other words, with probability $p_i = 1/(i + 1)$, the bullet path cuts segment $s_i$. The total expected number of segments cut by this bullet path is therefore

$$\sum_{i=1}^{k} p_i = \sum_{i=1}^{k} \frac{1}{i+1} \leq \sum_{i=1}^{k+1} \frac{1}{i} \leq O(\log(k + 1)) \leq O(\log n),$$

as desired.

![Figure 6: Solution to the Challenge Problem.](image-url)