PL Exercises

Question 1: Consider the trapezoid decomposition algorithm. With an example, show that without randomly shuffling the line segments, the following situations could arise.

- DS consumes $\Omega(n^2)$ space
- Query time becomes $\Omega(n)$

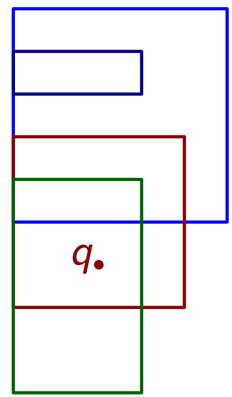
PL Exercises

Question 1: Consider the trapezoid decomposition algorithm. With an example, show that without randomly shuffling the line segments, the following situations could arise.

- DS consumes $\Omega(n^2)$ space
- Query time becomes $\Omega(n)$

Question 2: Probably, the simplest data structure one can consider is a linked list. Furthermore, it is clear that every element of a linked list has one incoming pointer and that we can insert and/or delete elements in a linked list by changing at most two pointers. Show that we can solve the following data structure problem.

- Input: A set of n "anchored" rectangles. The i-th rectangle is $[0, a_i] \times [b_i, c_i]$.
- Queries: A point (x, y) defined by two values x and y given at the query time.
- Output: The list of all the rectangles that contian the point (x, y).
- The data structure should use $O(n)$ space and should be able to produce the output in $O(\log n + k)$ time where k is the size of the output.

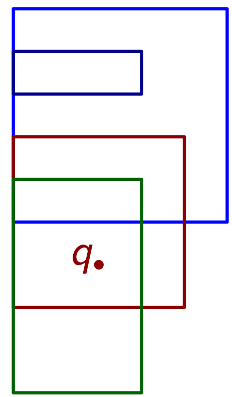


PL Exercises

Question 1: Consider the trapezoid decomposition algorithm. With an example, show that without randomly shuffling the line segments, the following situations could arise.

- DS consumes $\Omega(n^2)$ space
- Query time becomes $\Omega(n)$

Question 2: Probably, the simplest data structure one can consider is a linked list. Furthermore, it is clear that every element of a linked list has one incoming pointer and that we can insert and/or delete elements in a linked list by changing at most two pointers. Show that we can solve the following data structure problem.



- Input: A set of n “anchored” rectangles. The i-th rectangle is $[0, a_i] \times [b_i, c_i]$.
- Queries: A point (x, y) defined by two values x and y given at the query time.
- Output: The list of all the rectangles that contian the point (x, y).
- The data structure should use $O(n)$ space and should be able to produce the output in $O(\log n + k)$ time where k is the size of the output.

Question 3: Solve the following data structure problem.

- Input: A set of n lines L.
- Queries: A point (x, y) defined by two values x and y given at the query time.
- Output: The number of lines that pass below the point (x, y).
- The data structure should use $O(n^2)$ space and should answer queries in $O(\log n)$ time.

TrapezoidalMap(S)

1. Compute a bounding box, and initialize the decomposition $\mathcal{T}$ and a search structure $i\mathcal{D}$.

2. Compute a random permutation $s_1, .., s_n$ of the segments in S.

Invariant: $\mathcal{D}$ is a search structure for $\mathcal{T}$

3. **for** $i \leftarrow 1$ **to** n **do**

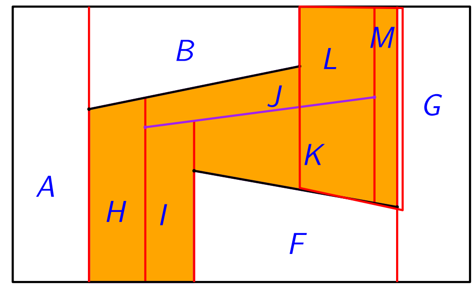
4. Find the trapezoids $\Delta_1, .., \Delta_k$ intersected by s_i.

5. Replace $\Delta_1, .., \Delta_k$ by new trapezoids.

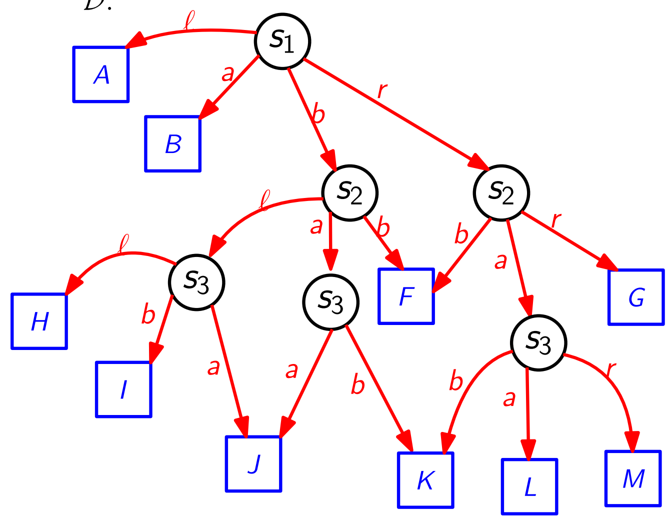
6. Update $\mathcal{D}$: remove the leaves for $\Delta_1, .., \Delta_k$, and insert inernal nodes containing s_i.

7. Place pointers from nodes containing s_i to the new trapezoids (and update DCEL)

$\mathcal{T}$:



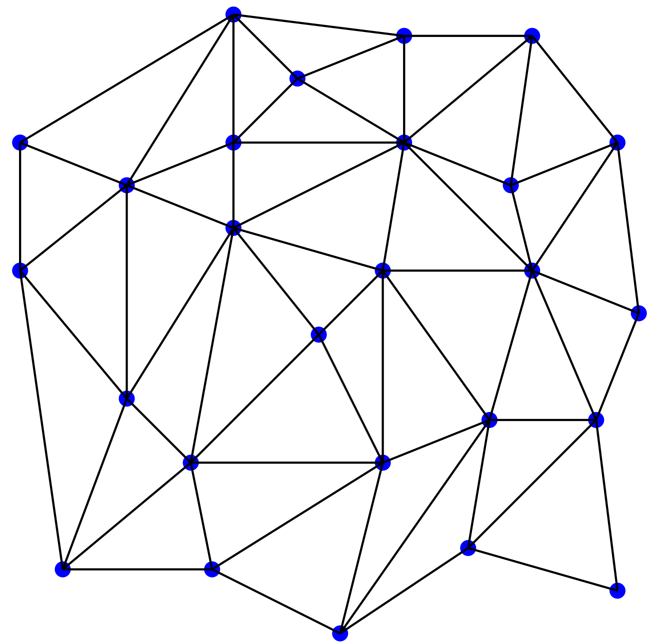
$\mathcal{D}$:



Question: How can we implement the step 5, where we merge some of some of the trapezoids created by the algorithm?

Another Solution: Kirkpatrick's

Input: A triangulation $\mathcal{T}$ with n vertices



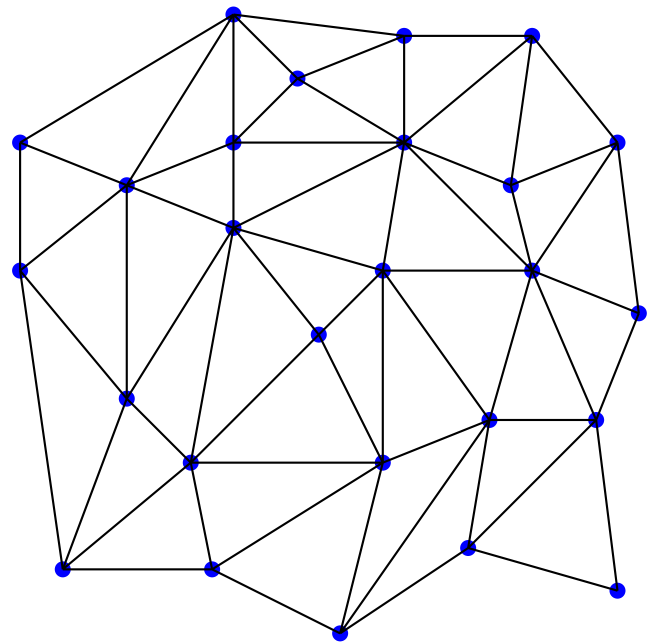
Another Solution: Kirkpatrick's

Input: A triangulation $\mathcal{T}$ with n vertices

We consider it a planar graph

We use a theorem that says, in a planar graph with V vertices, and E edges, we have:

$$E \leq 3V$$



Another Solution: Kirkpatrick's

Input: A triangulation $\mathcal{T}$ with n vertices

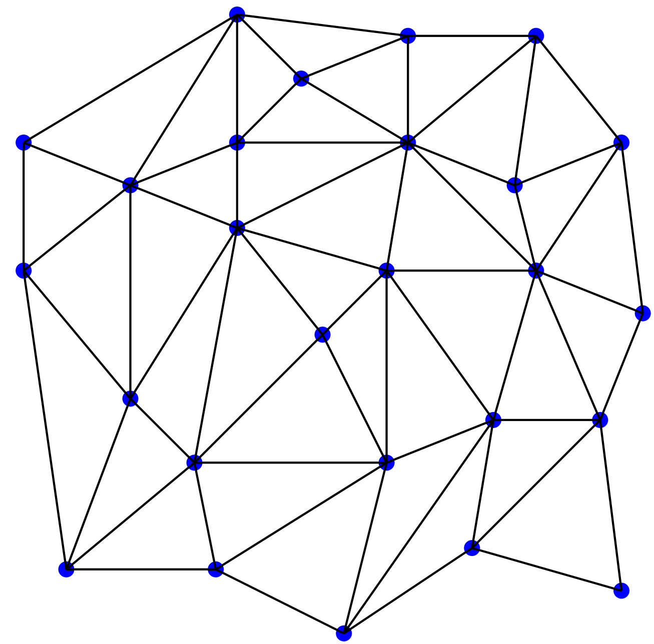
We consider it a planar graph

We use a theorem that says, in a planar graph with V vertices, and E edges, we have:

$$E \leq 3V$$

Question 1: Show that there exists a subset A of vertices:

1. Every vertex in A has degree at most 11
2. $|A| \geq \frac{n}{2}$



Another Solution: Kirkpatrick's

Input: A triangulation $\mathcal{T}$ with n vertices

We consider it a planar graph

We use a theorem that says, in a planar graph with V vertices, and E edges, we have:

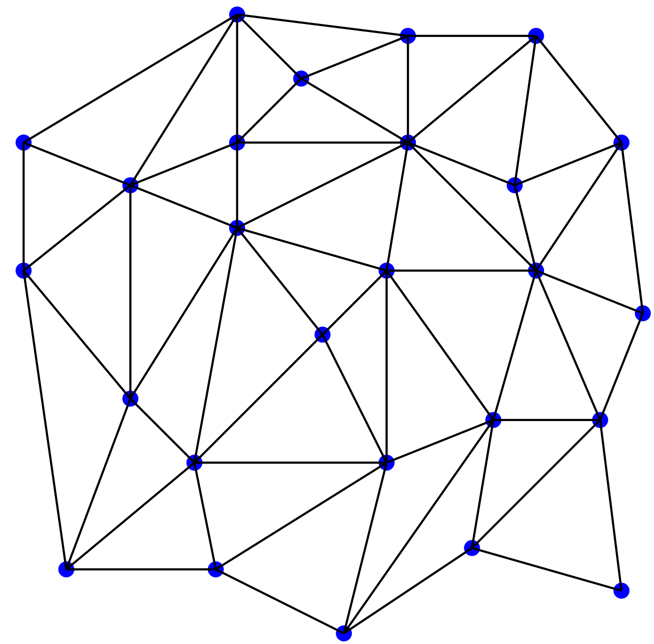
$$E \leq 3V$$

Question 1: Show that there exists a subset A of vertices:

1. Every vertex in A has degree at most 11
2. $|A| \geq \frac{n}{2}$

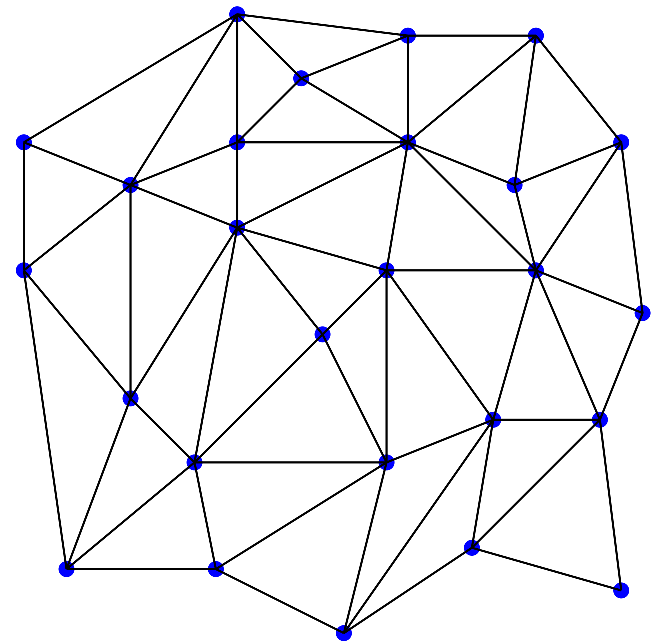
Question 2: Show that there exists a set $I \subset A$:

1. Every vertex in I has degree at most 11
2. $|I| \geq \frac{n}{24}$
3. No two vertcies in I are connected



Another Solution: Kirkpatrick's

Input: A triangulation $\mathcal{T}$ with n vertices



We consider it a planar graph

We use a theorem that says, in a planar graph with V vertices, and E edges, we have:

$$E \leq 3V$$

Question 1: Show that there exists a subset A of vertices:

1. Every vertex in A has degree at most 11
2. $|A| \geq \frac{n}{2}$

Question 2: Show that there exists a set $I \subset A$:

1. Every vertex in I has degree at most 11
2. $|I| \geq \frac{n}{24}$
3. No two vertcies in I are connected

Question 3: Let v be a vertex of degree d in $\mathcal{T}$. Let $\mathcal{T}'$ be the triangulation obtained by deleting v and then "re-triangulating" the resulting hole.
Show that if we know the triangle containing a query point q in $\mathcal{T}'$, then we can compute the triangle containg q in $\mathcal{T}$ in $O(d)$ time.

Another Solution: Kirkpatrick's

Input: A triangulation $\mathcal{T}$ with n vertices

We consider it a planar graph

We use a theorem that says, in a planar graph with V vertices, and E edges, we have:

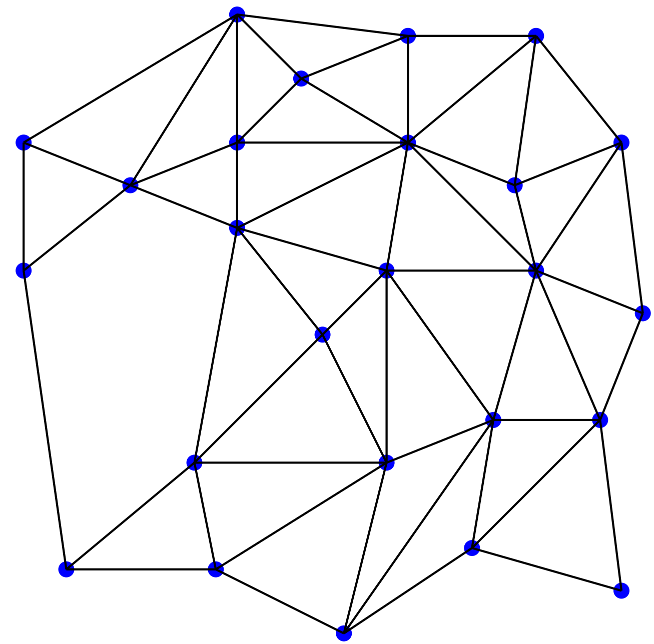
$$E \leq 3V$$

Question 1: Show that there exists a subset A of vertices:

1. Every vertex in A has degree at most 11
2. $|A| \geq \frac{n}{2}$

Question 2: Show that there exists a set $I \subset A$:

1. Every vertex in I has degree at most 11
2. $|I| \geq \frac{n}{24}$
3. No two vertcies in I are connected



Question 3: Let v be a vertex of degree d in $\mathcal{T}$. Let $\mathcal{T}'$ be the triangulation obtained by deleting v and then "re-triangulating" the resulting hole.
Show that if we know the triangle containing a query point q in $\mathcal{T}'$, then we can compute the triangle containg q in $\mathcal{T}$ in $O(d)$ time.

Another Solution: Kirkpatrick's

Input: A triangulation $\mathcal{T}$ with n vertices

We consider it a planar graph

We use a theorem that says, in a planar graph with V vertices, and E edges, we have:

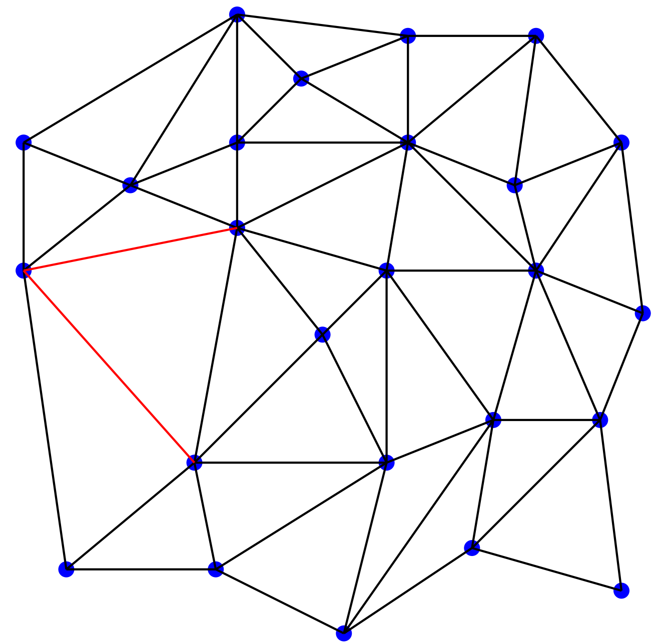
$$E \leq 3V$$

Question 1: Show that there exists a subset A of vertices:

1. Every vertex in A has degree at most 11
2. $|A| \geq \frac{n}{2}$

Question 2: Show that there exists a set $I \subset A$:

1. Every vertex in I has degree at most 11
2. $|I| \geq \frac{n}{24}$
3. No two vertcies in I are connected



Question 3: Let v be a vertex of degree d in $\mathcal{T}$. Let $\mathcal{T}'$ be the triangulation obtained by deleting v and then "re-triangulating" the resulting hole.
Show that if we know the triangle containing a query point q in $\mathcal{T}'$, then we can compute the triangle containg q in $\mathcal{T}$ in $O(d)$ time.

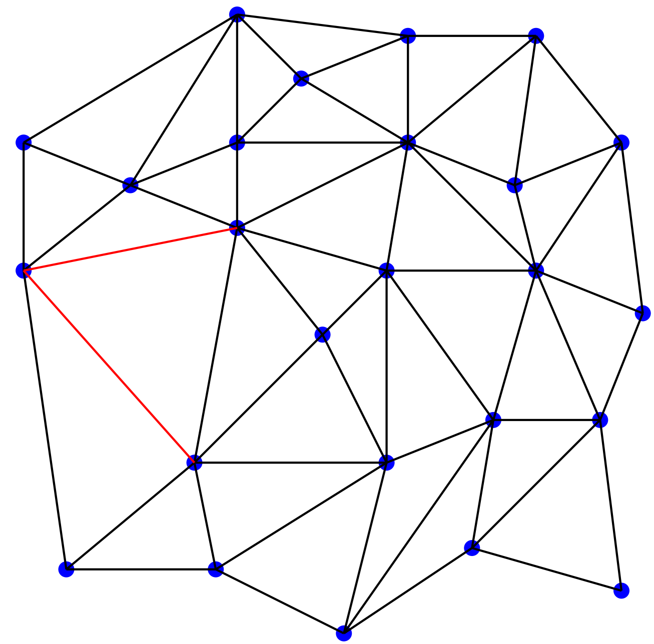
Another Solution: Kirkpatrick's

Input: A triangulation $\mathcal{T}$ with n vertices

We consider it a planar graph

We use a theorem that says, in a planar graph with V vertices, and E edges, we have:

$$E \leq 3V$$



Question 1: Show that there exists a subset A of vertices:

1. Every vertex in A has degree at most 11
2. $|A| \geq \frac{n}{2}$

Question 2: Show that there exists a set $I \subset A$:

1. Every vertex in I has degree at most 11
2. $|I| \geq \frac{n}{24}$
3. No two vertcies in I are connected

Question 3: Let v be a vertex of degree d in $\mathcal{T}$. Let $\mathcal{T}'$ be the triangulation obtained by deleting v and then "re-triangulating" the resulting hole.
Show that if we know the triangle containing a query point q in $\mathcal{T}'$, then we can compute the triangle containg q in $\mathcal{T}$ in $O(d)$ time.

Question 4: Use the answers to question 2& 3 and recursion, to build a data structure that uses $O(n)$ space and it can answer queries in $O(\log n)$ time.
The recursion should give you a recursive formula for the query time and the space usage of the data structure. Use Master's theorem to analyse the space and query time.