So, our concern is to keep $O(\log n)$ query time but reduce pre-processing time/space to $O(n)$

In fact we will do this for arbitrary planar straight-line graphs.

more general more interesting
First pre-processing step: triangulate (each region) 

$O(n)$ time* & space

we will actually determine which triangle contains any given query point

even more general

* $O(n)$ time is possible if the graph is connected: every face has a weakly simple polygonal boundary

If the graph is disconnected then it can take $\sim n\log n$ time to triangulate
Even more general: create a triangular outer face

All of this is still $O(n)$ time & space
Recall that we used the Euler formula \( V - E + F = 2 \) to show that a triangulation has

\[
e = 3n - h - 3 \quad \text{or} \quad e = 3n - 6
\]

when outer face is a triangle.

\( V - E + F = 2 \) applies to any connected planar graph (in fact, to convex polyhedra) by projection.

**Induction on faces:**
- \( F = 1 \): tree. \( E = V - 1 \)
- \( F > 1 \):
  - Remove an edge between 2 faces.
  - Remains connected. \( F \rightarrow F - 1 \) \( E \rightarrow E - 1 \)

**Induction on vertices**
- \( V = 1 \):
  - Only loops \( F = E + 1 \)
- \( V > 1 \):
  - Contract edge \( x \neq y \)
  - \( V \rightarrow V - 1 \) \( E \rightarrow E - 1 \)

**Induction on edges**
- \( E = 0 \):
  - One vertex, one face
- \( E > 1 \):
  - if \( x \neq y \) contract as before
  - else remove as before
  - \( E \rightarrow E - 1 \) \( F \rightarrow F - 1 \) or \( V \rightarrow V - 1 \)
What is the average degree of a triangulation?

\[ \frac{1}{n} \cdot \sum_{i=1}^{n} d(v_i) = \frac{1}{n} \cdot 2e = \frac{6n-12}{n} \leq 6 \]

\[ \implies \text{Every triangulation has a vertex w/ degree } \leq 5 \]

(but might only have 12?)

Can we find many low-degree vertices? \[ \rightarrow \text{not if "low" = 5. what if "low" = 8?} \]

Say you had \( \gg \frac{n}{2} \) vertices w/ degree \( \gg 9 \)

\[ \sum_{d \gg 9} d(v_i) \gg 9 \cdot \frac{n}{2} \]

sum degrees of \( \frac{n}{2} \) of them

\[ \implies \sum d(v_i) \gg 3 \cdot \frac{n}{2} \]

\( \Sigma + \Sigma \gg 6n \) contradiction

always have \( \gg \frac{n}{2} \) w/ deg. \( \leq 8 \)
Always has $\geq \frac{n}{2}$ vertices w/ degree $\leq 8$

In fact many of them must be nicely separated

independent set $\leftarrow$

no 2 are neighbors

Why? $\rightarrow$

start w/ $\frac{n}{2}$ of them; pick one, and mark it.
Mark all of its neighbors too (don't care about their deg.)

$\Rightarrow$ now we've marked $\leq 9$ vertices

Repeat: pick any unmarked vertex w/ deg. $\leq 8$, & mark

We can do this $\frac{n/2}{9}$ times

$\therefore$ get $\frac{n}{18}$ independent vertices w. degree $\leq 8$
what matters is we find $\circ$ w/ deg. $\leq 8$ and then $\circ$ also w/ deg. $\leq 8$ and not a neighbor
Time to find independent set of degree \( \leq 8 \)?

- First mark all degree \( \geq 9 \): \( O(n) \) \((\exists \text{d}(v_i) = O(n))\)
- Place all unmarked vertices in a list
- Each time scan from start of list for first unmarked vertex (& delete marked)
- When marking neighbors can also delete from list

Requires simple graph structure (access to neighbors) & links to the list.

\( O(n) \) overall
Every iteration: get rid of constant fraction of points

: re-triangulate: $O(1)$ per hole

holes have size $O(1)$ so do each one with any slow algorithm

$O(n)$ overall
Search:
(point location)

Look at top level: \( \text{outer triangle} \)

Then find triangles of next level that overlap

At every level, you locate one triangle \( \rightarrow \) and in next level \( O(1) \) overlap it

(8 queries)
We are done after $O(\log n)$ levels because we eliminate a constant fraction of the nodes each time.

Notice that the query region can expand, but it's always constrained to be the union of 8 triangles.