Delaunay Triangulations of Imprecise Points in Linear Time after Preprocessing

Maarten Löffler

Utrecht University

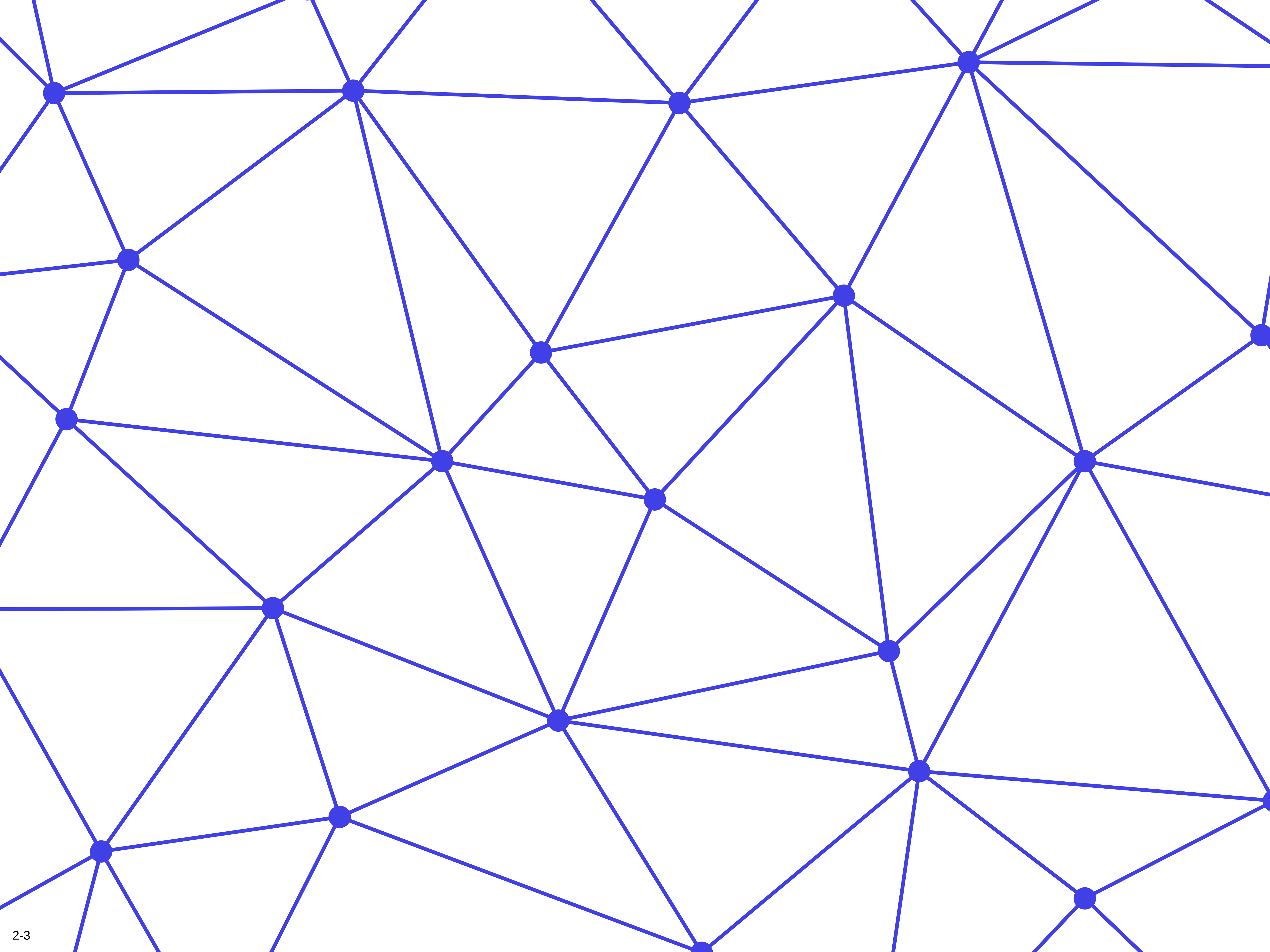
the Netherlands

Jack Snoeyink

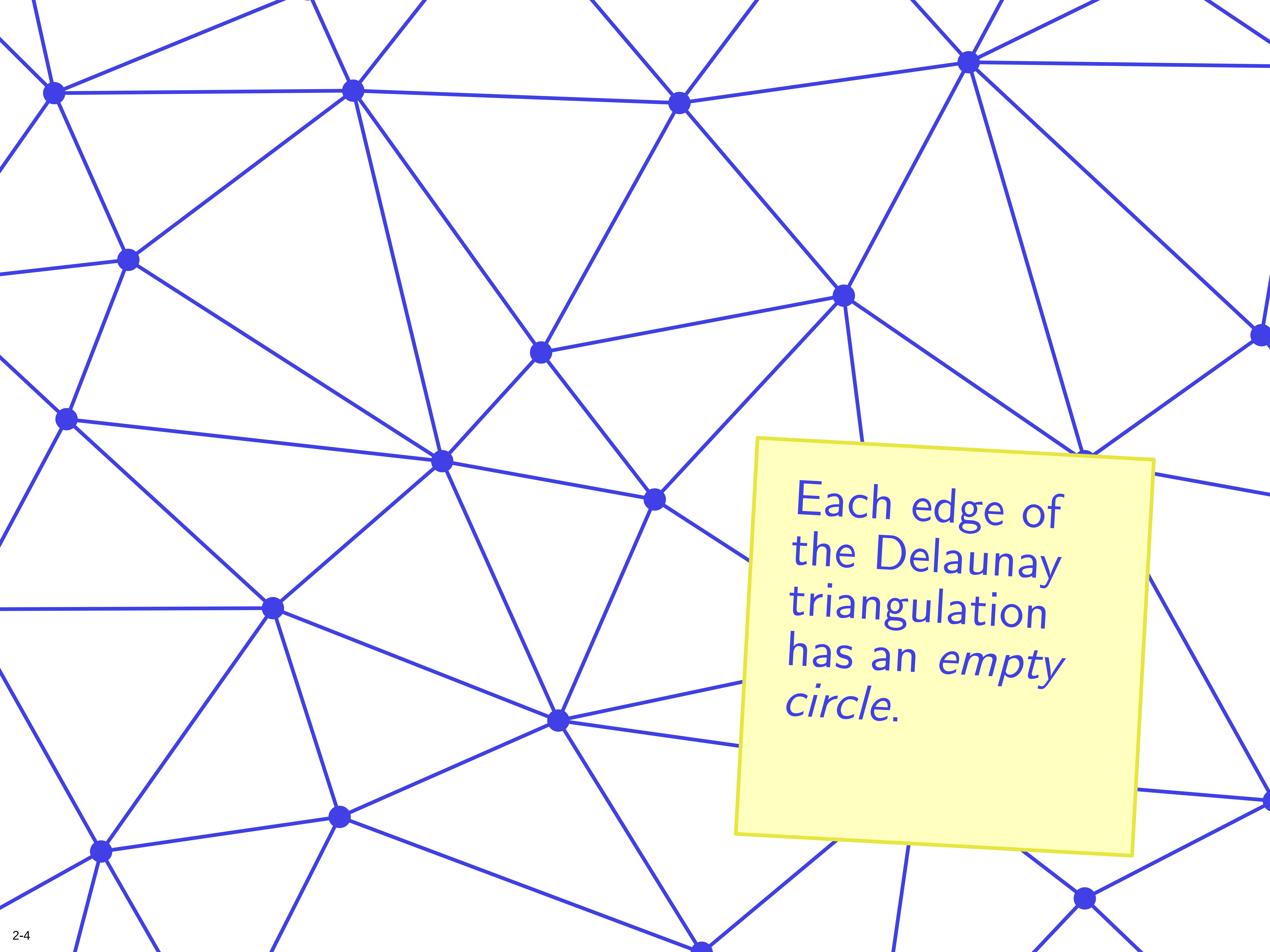
UNC Chapel Hill

United States

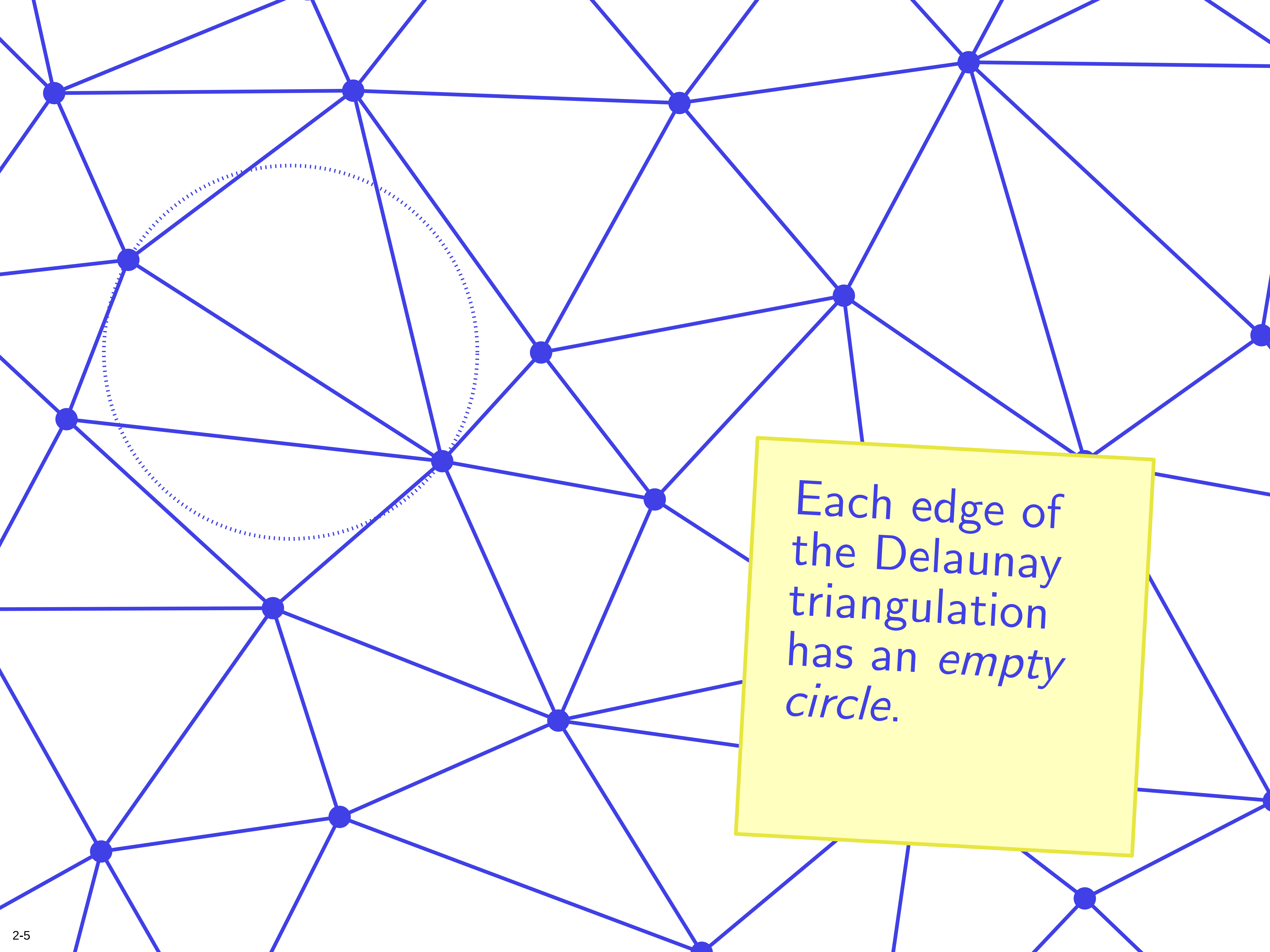
What were *Delaunay triangulations* again?



2-3



Each edge of the Delaunay triangulation has an *empty circle.*



Each edge of the Delaunay triangulation has an *empty circle.*

So, what exactly are *imprecise points*?

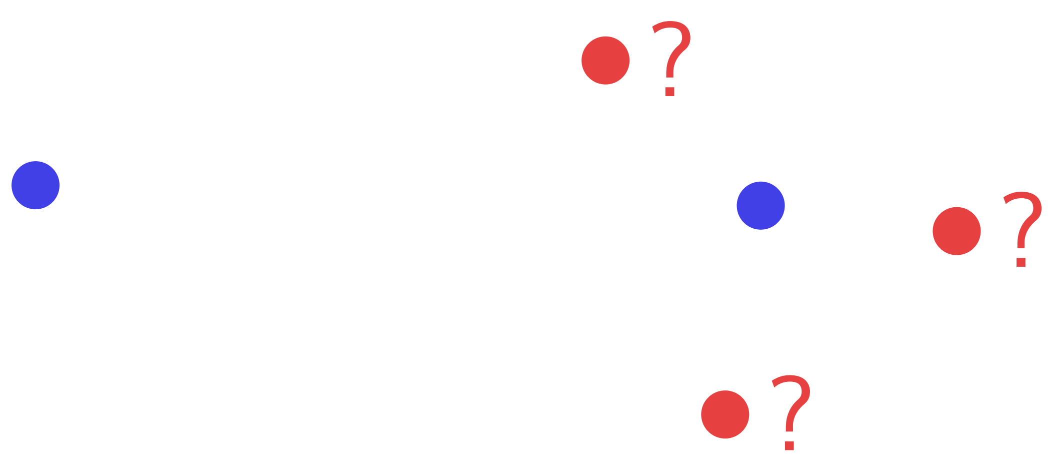
In traditional computational geometry, input points are presumed to be precise.

In traditional computational geometry, input points are presumed to be precise.

However, in practical applications locations of input points are *not* precise.

In traditional computational geometry, input points are presumed to be precise.

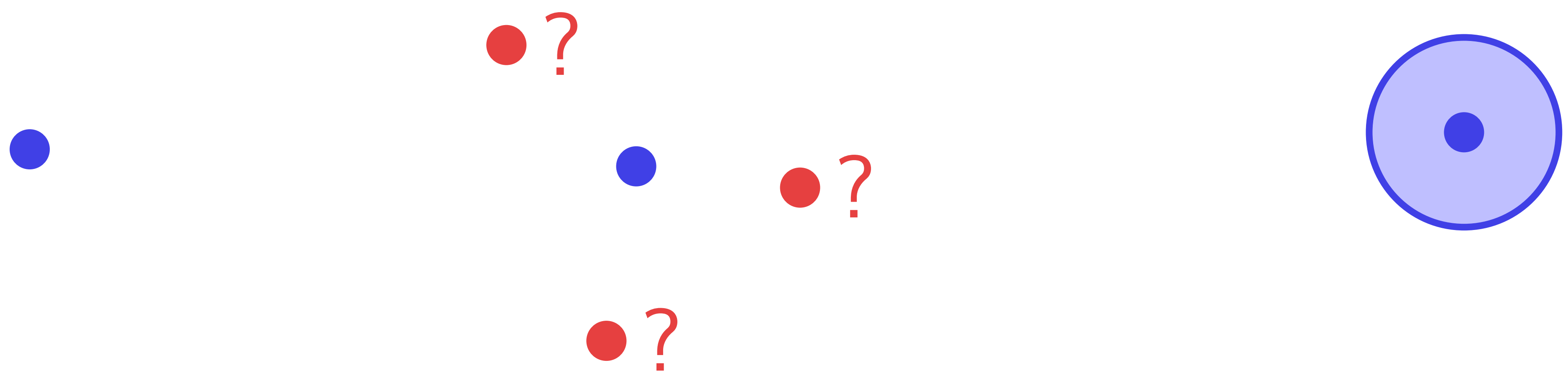
However, in practical applications locations of input points are *not* precise.



In traditional computational geometry, input points are presumed to be precise.

However, in practical applications locations of input points are *not* precise.

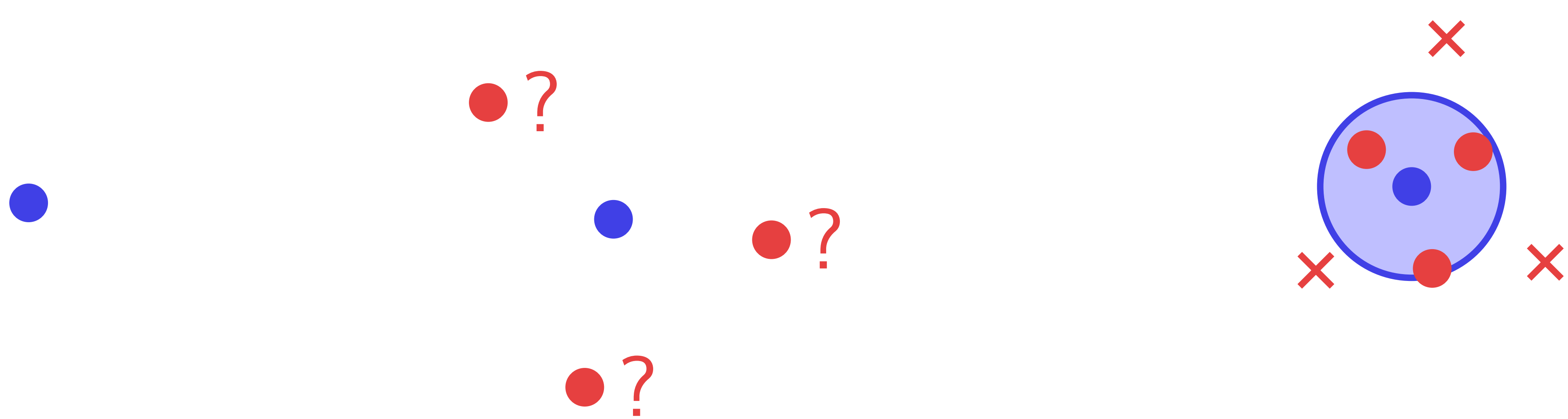
Often, a bound is available: a point is known to be at most ε away from a given location.



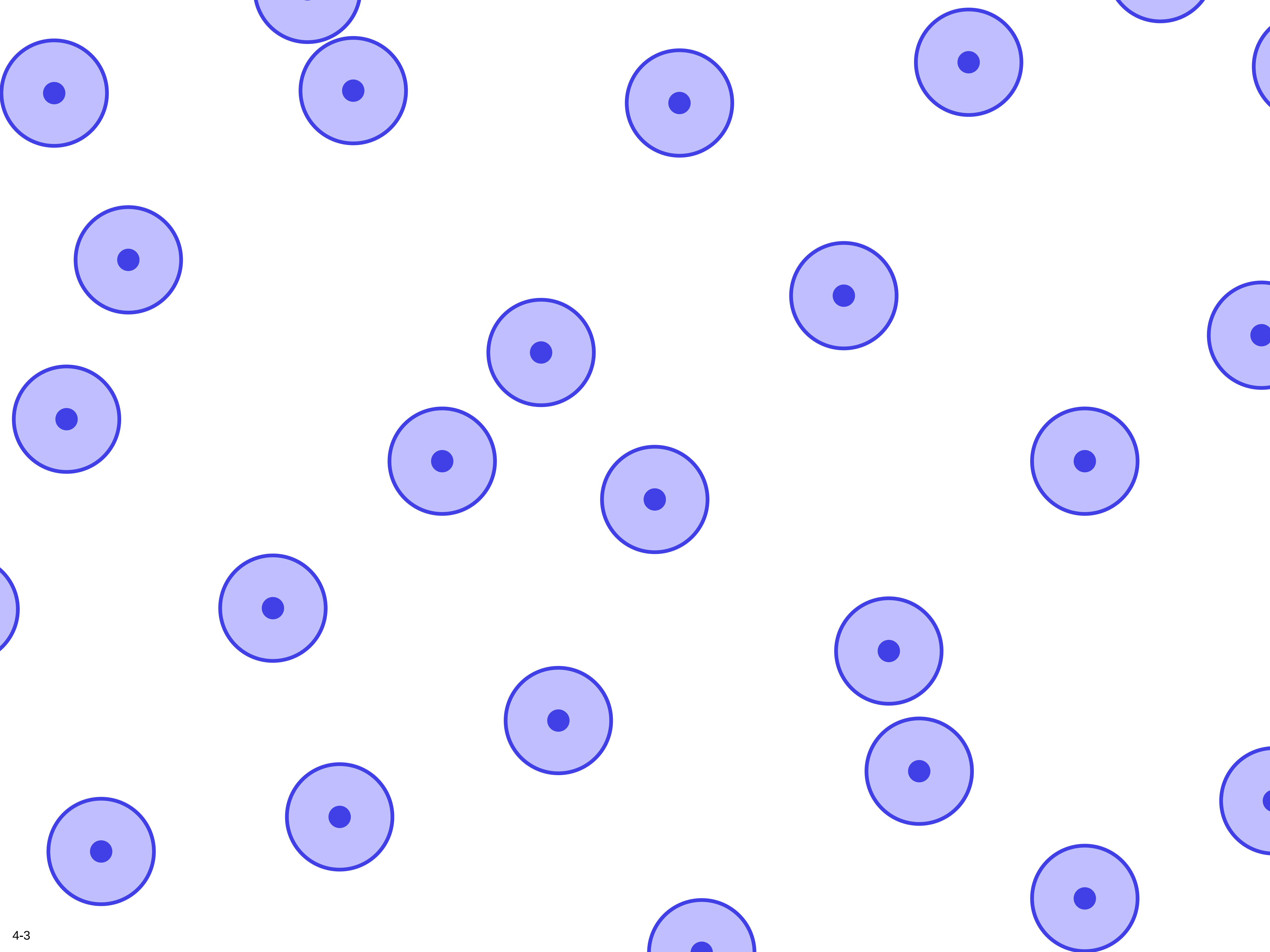
In traditional computational geometry, input points are presumed to be precise.

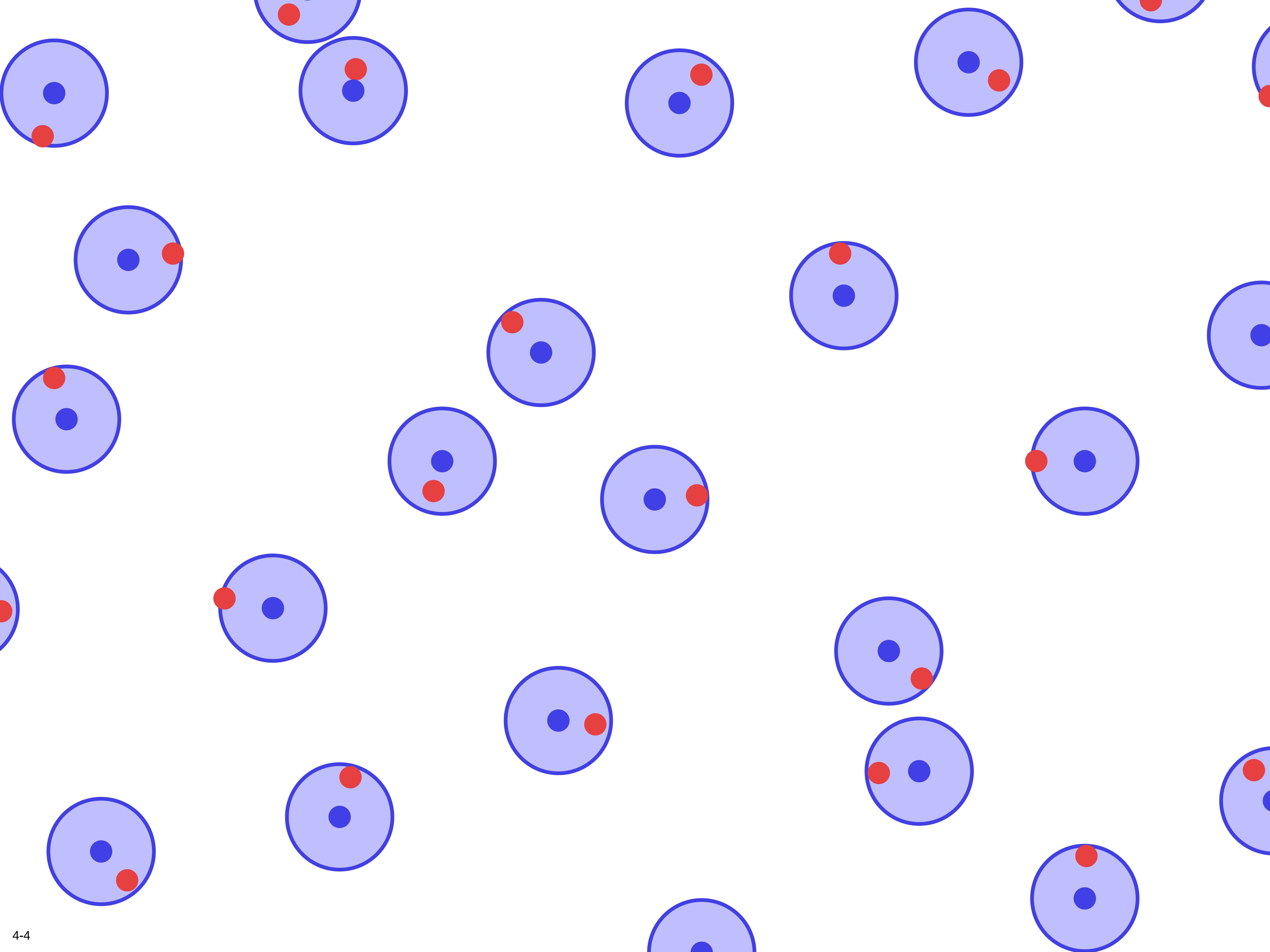
However, in practical applications locations of input points are *not* precise.

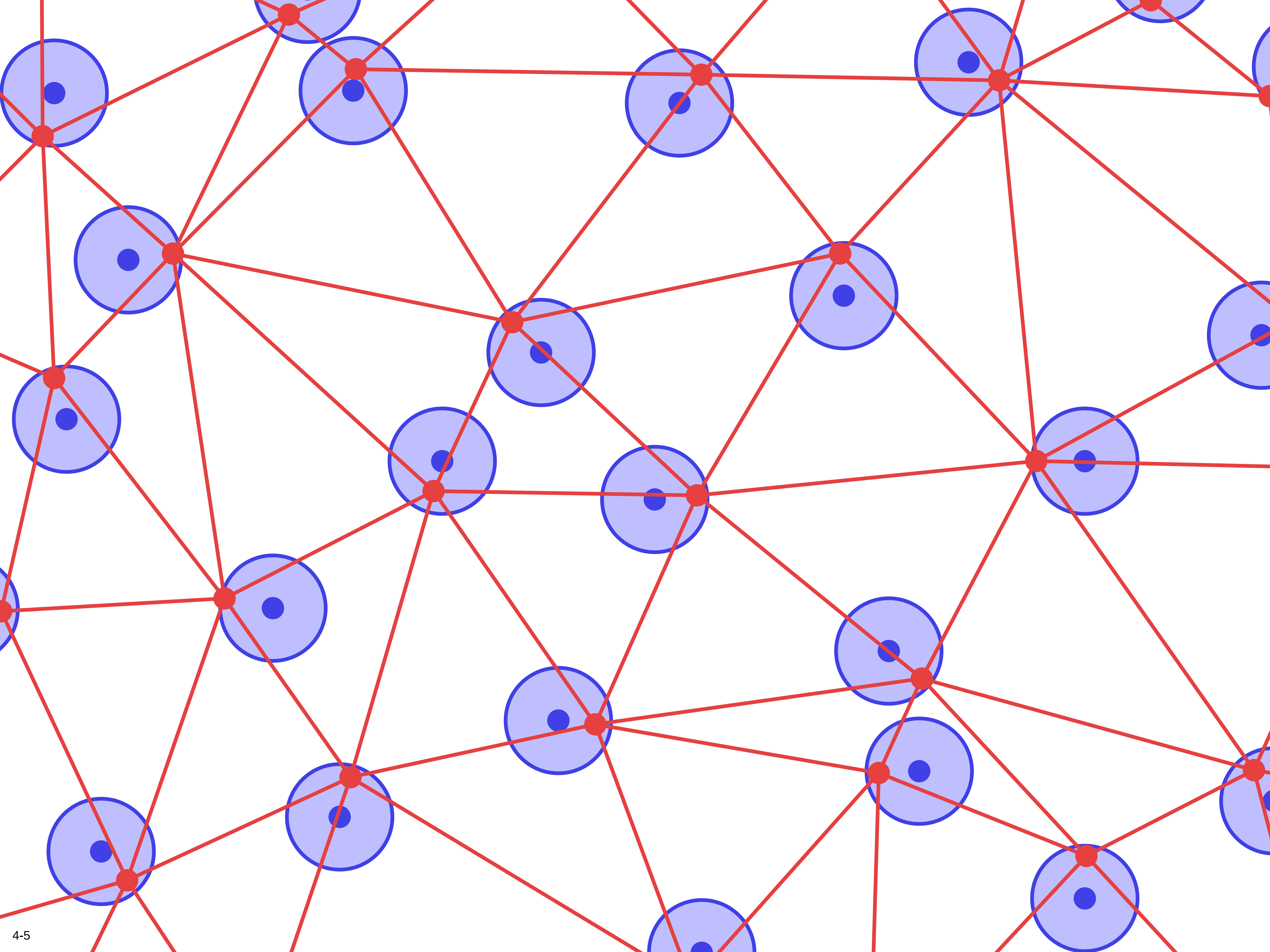
Often, a bound is available: a point is known to be at most ε away from a given location.

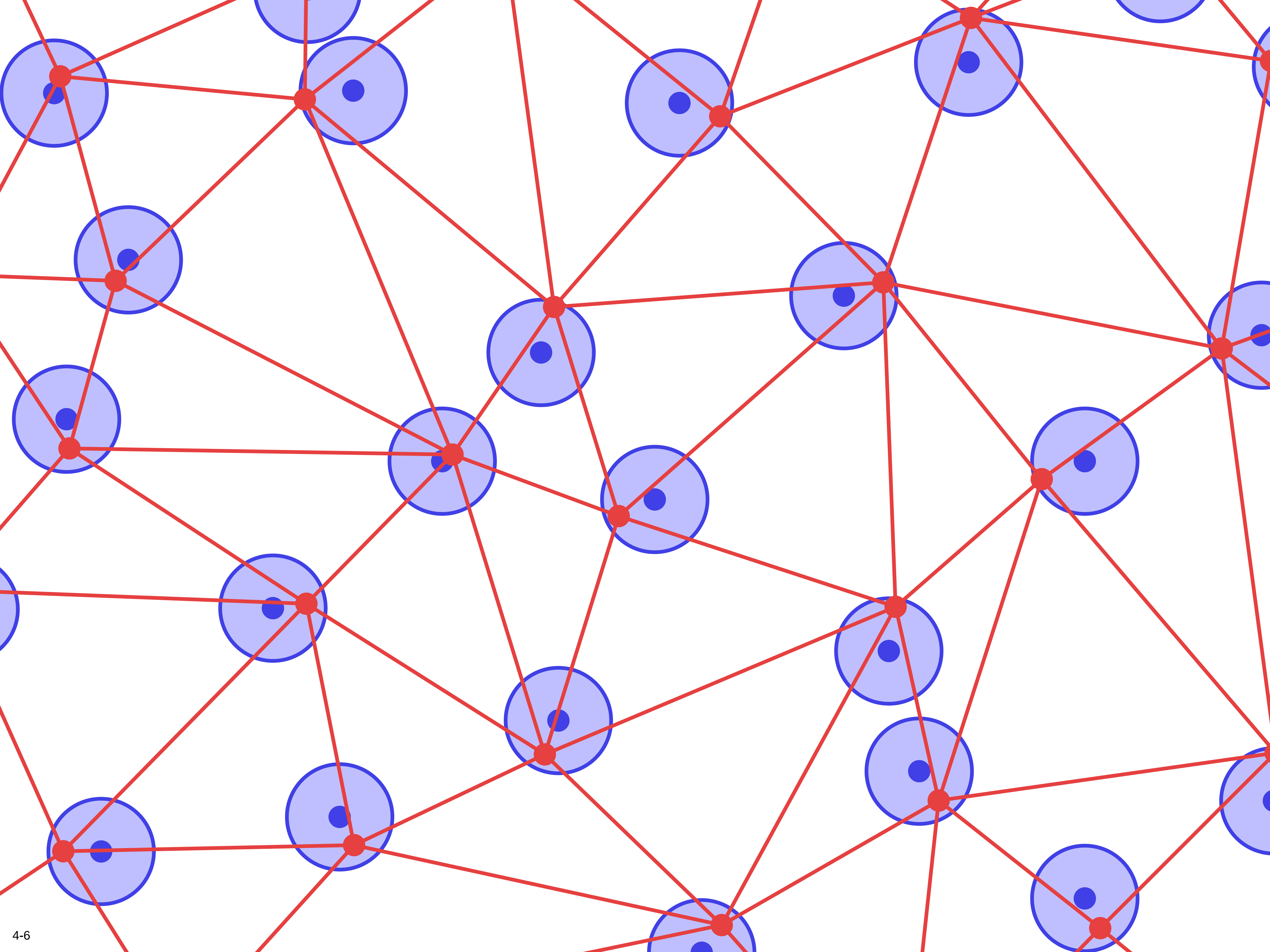


Then, what is the Delaunay triangulation of imprecise points?

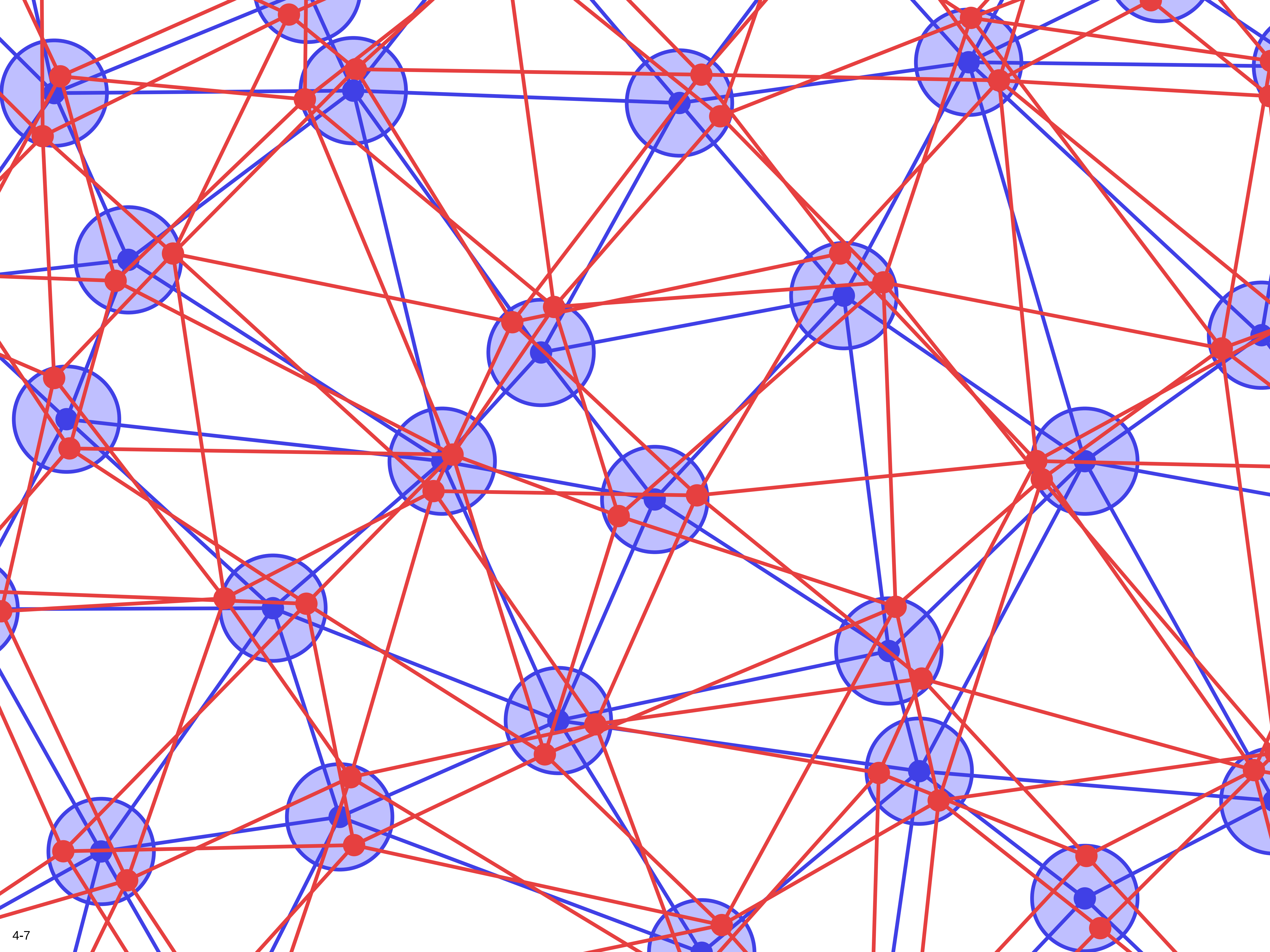


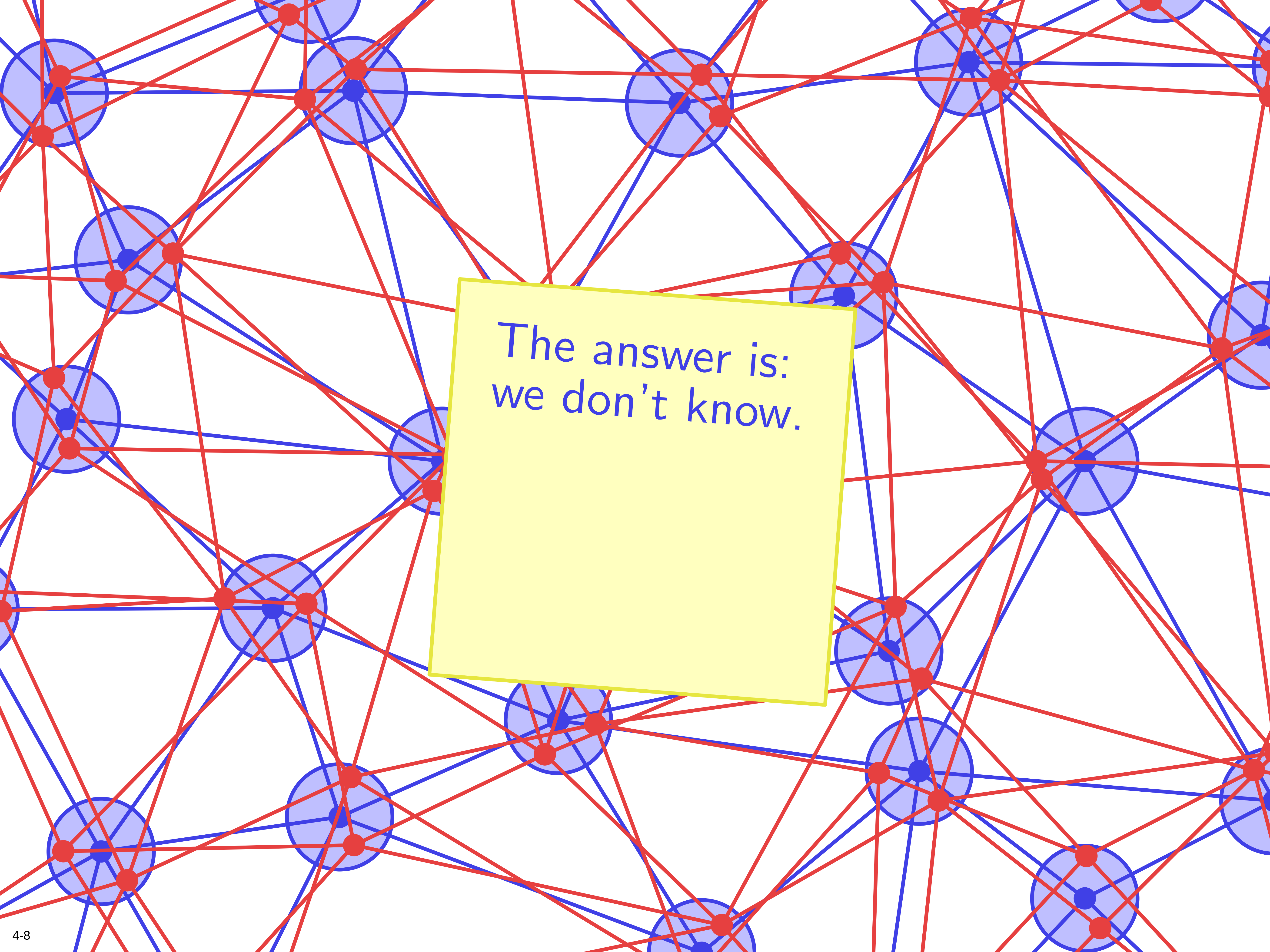






4-6


4-7



The answer is:
we don't know.

So, we have imprecise input data.

What *can* we do?

Change the problem: work on regions. [Goodrich & Mitchell & Orletsky 1994]

Change the problem: work on regions. [Goodrich & Mitchell & Orletsky 1994]

Compute only *certain* parts of the output. [Khanban 2005]

Change the problem: work on regions. [Goodrich & Mitchell & Orletsky 1994]

Compute only *certain* parts of the output. [Khanban 2005]

Compute *all* possible output features. [Bandyopad-hyay & Snoeyink 2007]

Change the problem: work on regions. [Goodrich & Mitchell & Orletsky 1994]

Compute only *certain* parts of the output. [Khanban 2005]

Compute *all* possible output features. [Bandyopadhyay & Snoeyink 2007]

Compute imprecision of result. [Löffler & van Kreveld 2008]

Change the problem: work on regions. [Goodrich & Mitchell & Orletsky 1994]

Compute only *certain* parts of the output. [Khanban 2005]

Compute *all* possible output features. [Bandyopadhyay & Snoeyink 2007]

Compute imprecision of result. [Löffler & van Kreveld 2008]

Compute imprecision for which result is certain. [Abellanas & Hurtado & Ramos 1999]

Change the problem: work on regions. [Goodrich & Mitchell & Orletsky 1994]

Compute only *certain* parts of the output. [Khanban 2005]

Compute *all* possible output features. [Bandyopadhyay & Snoeyink 2007]

Compute imprecision of result. [Löffler & van Kreveld 2008]

Compute imprecision for which result is certain. [Abellanas & Hurtado & Ramos 1999]

Make output depend on imprecision. [Guibas & Salesin & Stolfi 1993]

Change the problem: work on regions. [Goodrich & Mitchell & Orletsky 1994]

Compute only *certain* parts of the output. [Khanban 2005]

Compute *all* possible output features. [Bandyopad-hyay & Snoeyink 2007]

Compute imprecision of result. [Löffler & van Kreveld 2008]

Compute imprecision for which result is certain. [Abellanas & Hurtado & Ramos 1999]

Compute exact output on input sampled from regions.

Make output depend on imprecision. [Guibas & Salesin & Stolfi 1993]

Compute only *certain* parts of the output. [Khanban 2005]

Change the problem: work on regions. [Goodrich & Mitchell & Orletsky 1994]

Compute *all* possible output features. [Bandyopad-hyay & Snoeyink 2007]

Then, what is the value of the information about the regions that we already have?

Compute imprecision for which result is certain. [Abellanas & Hurtado & Ramos 1999]

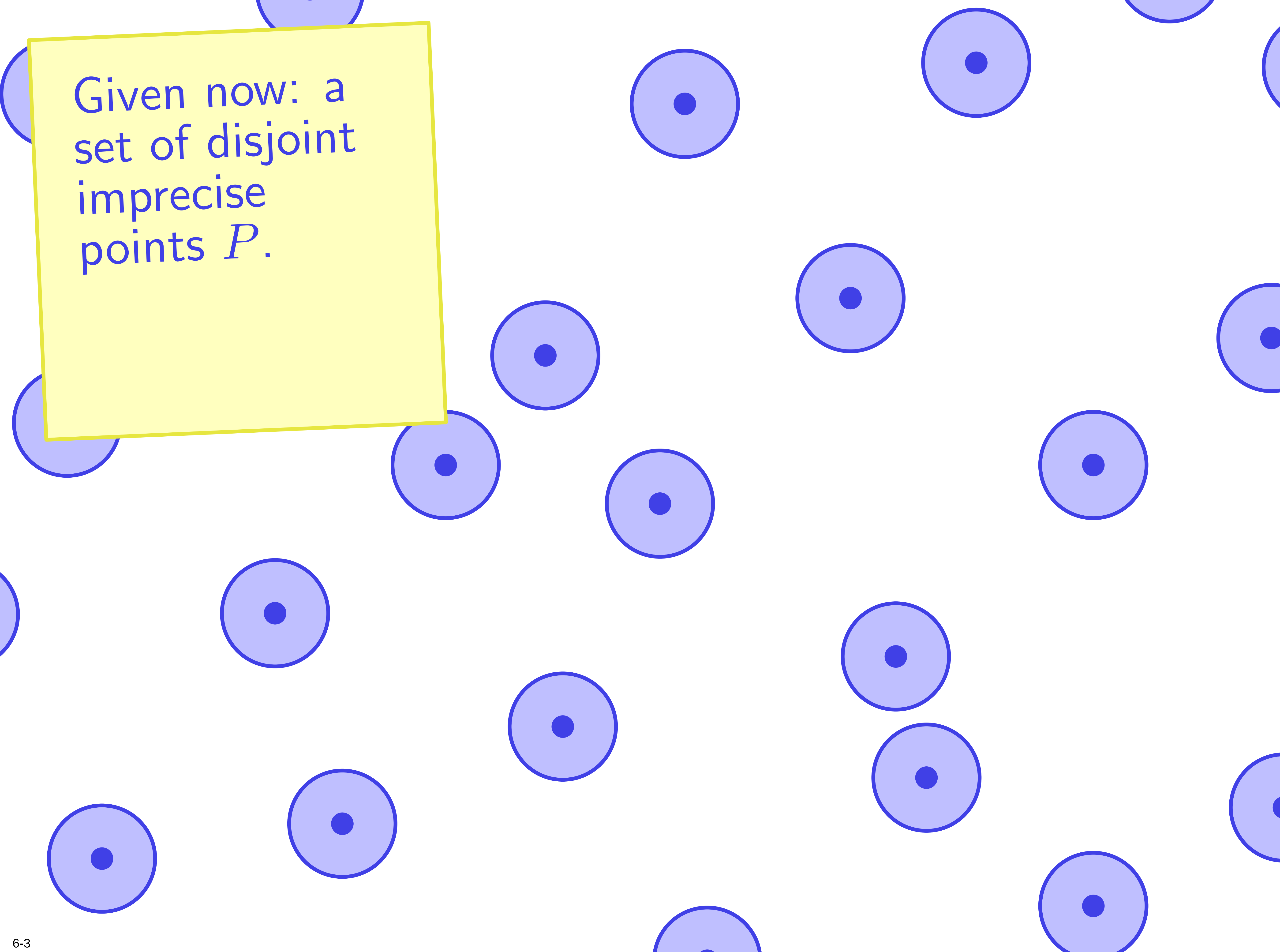
Compute exact output on input sampled from regions.

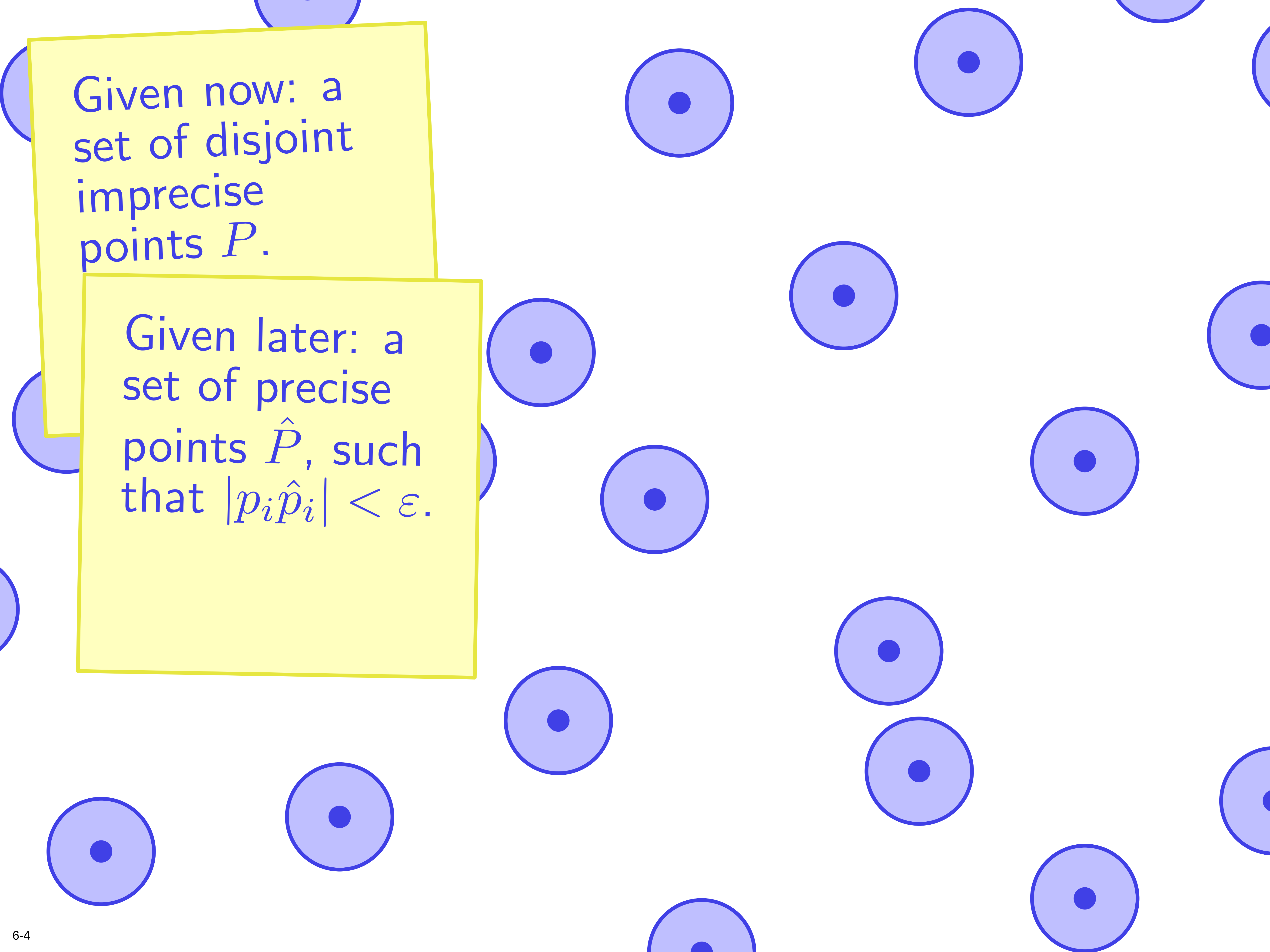
Make output depend on imprecision. [Guibas & Salesin & Stolfi 1993]

Let's define a *problem statement*.

Given now: a set of disjoint imprecise points P.

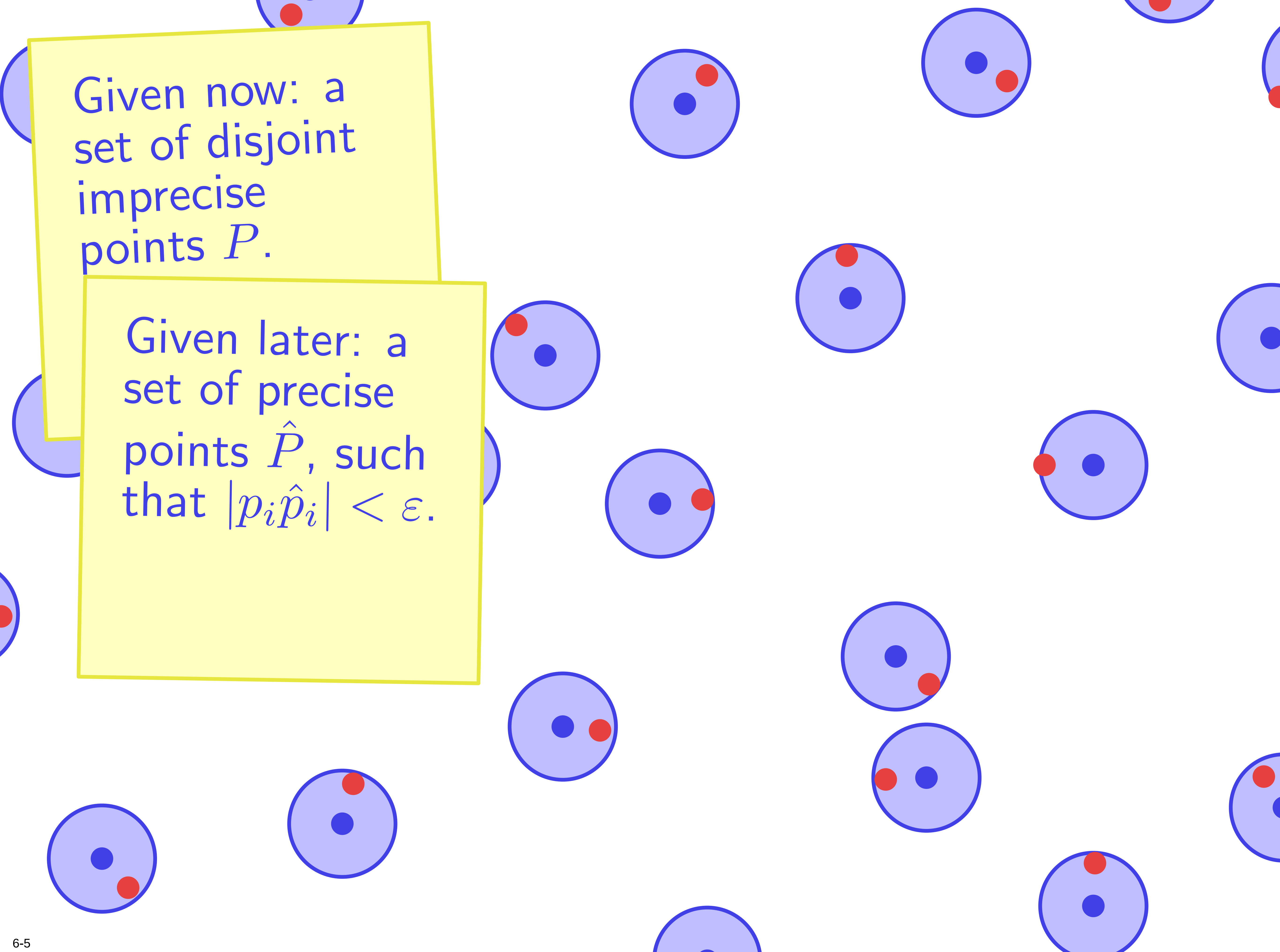
Given now: a set of disjoint imprecise points P.





Given now: a set of disjoint imprecise points P.

Given later: a set of precise points $\hat{P}$, such that $|p_i \hat{p}_i| < \varepsilon$.

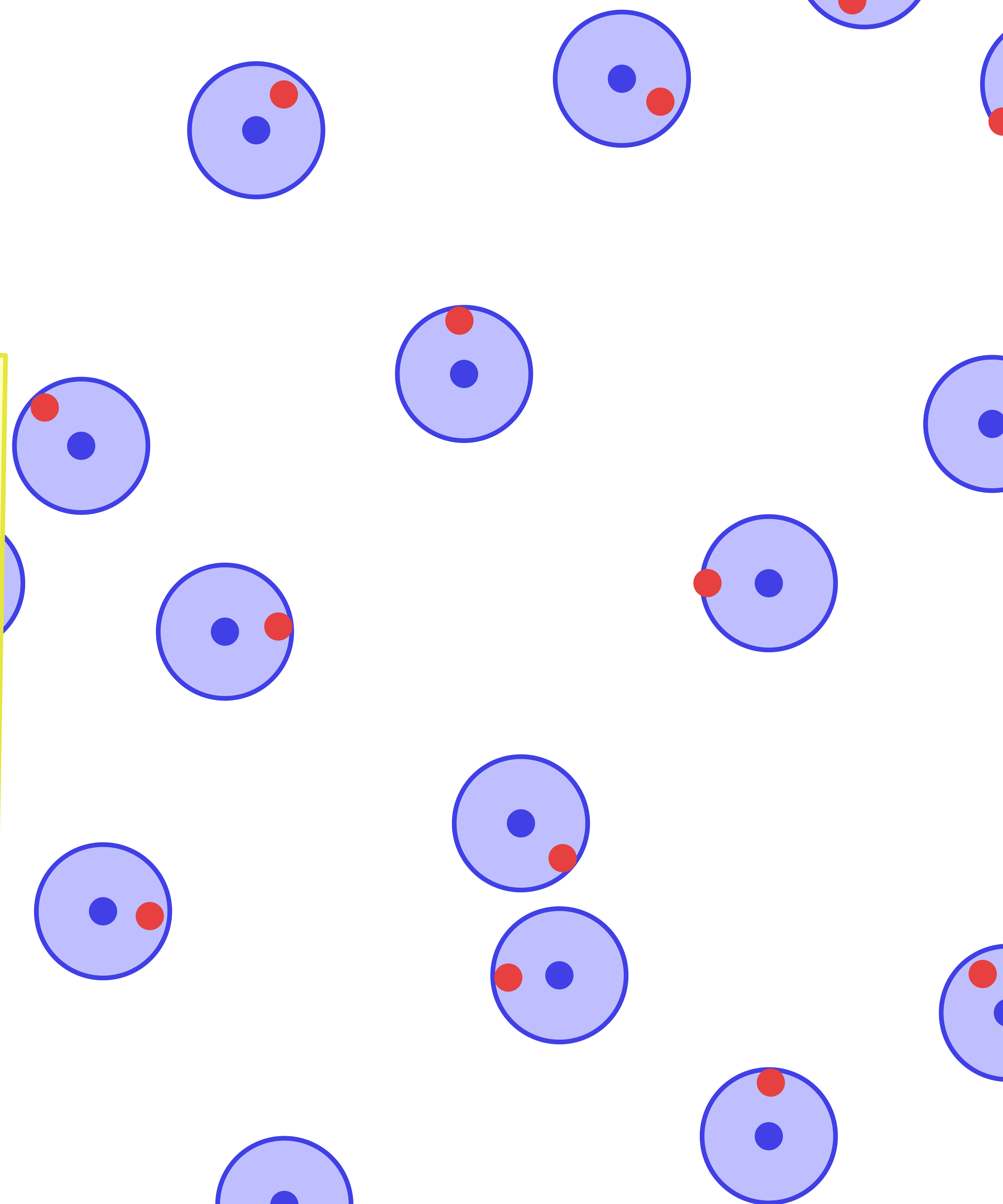

Given now: a set of disjoint imprecise points P.
Given later: a set of precise points $\hat{P}$, such that $|p_i \hat{p}_i| < \varepsilon$.

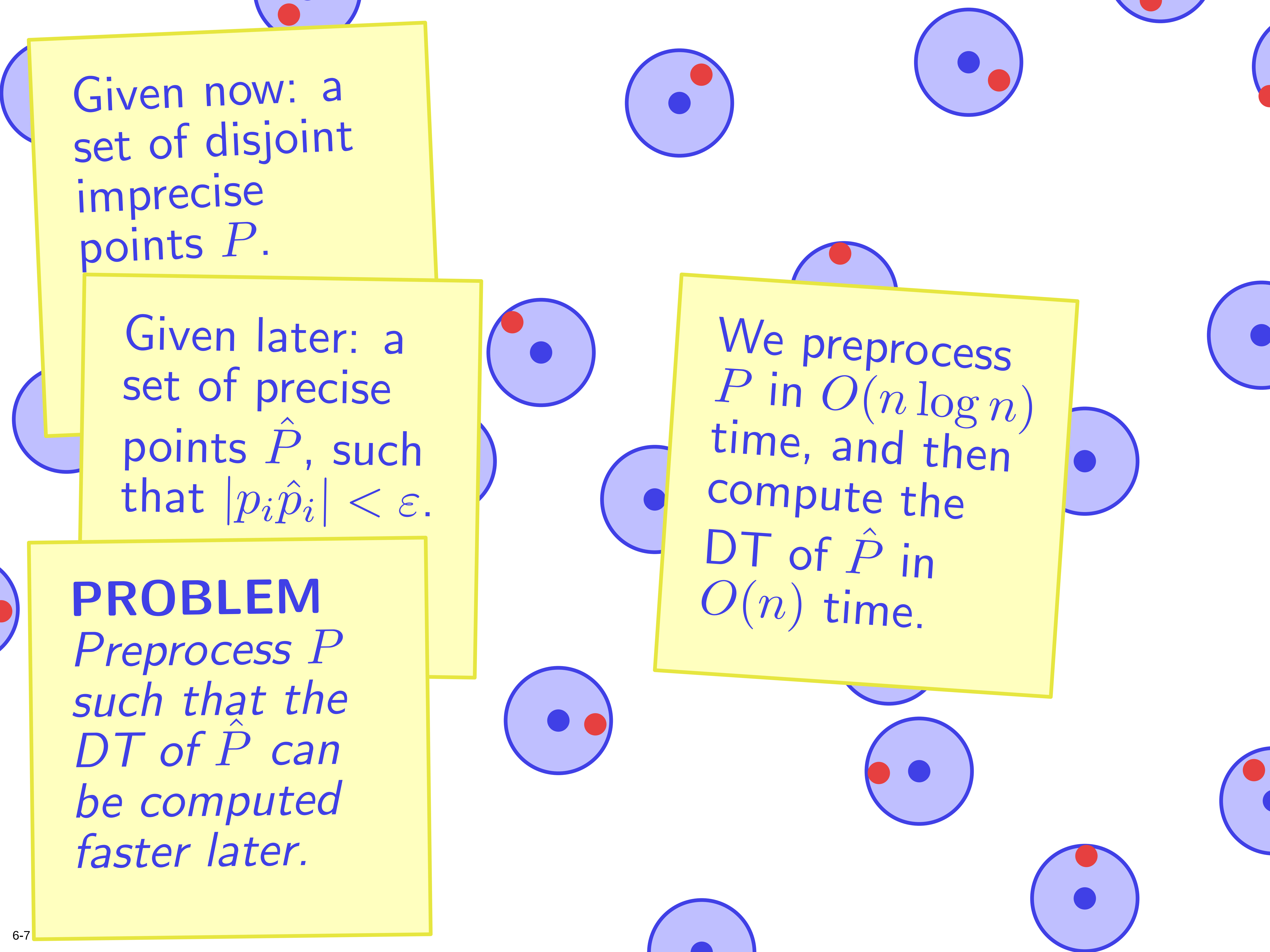
Given now: a set of disjoint imprecise points P.

Given later: a set of precise points $\hat{P}$, such that $|p_i\hat{p}_i| < \varepsilon$.

PROBLEM

Preprocess P such that the DT of $\hat{P}$ can be computed faster later.





Given now: a set of disjoint imprecise points P.

Given later: a set of precise points $\hat{P}$, such that $|p_i\hat{p}_i| < \varepsilon$.

PROBLEM
Preprocess P such that the DT of $\hat{P}$ can be computed faster later.

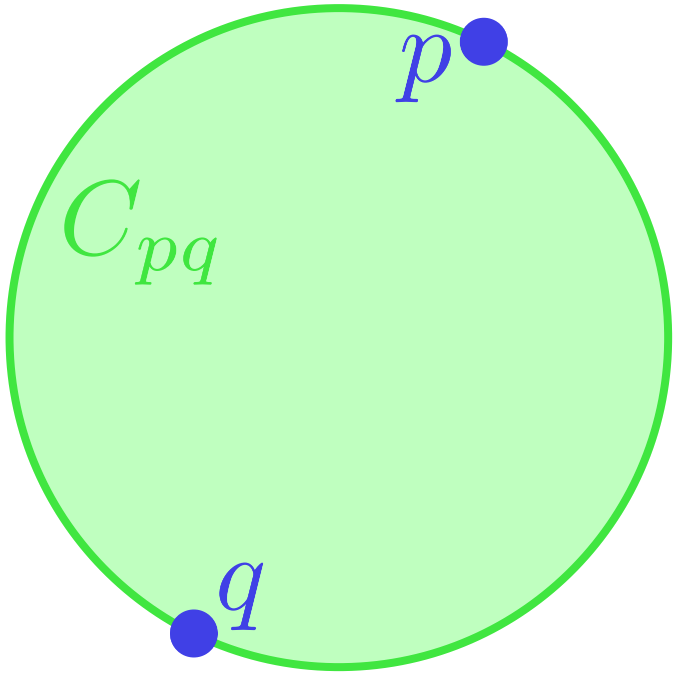
We preprocess P in $O(n \log n)$ time, and then compute the DT of $\hat{P}$ in $O(n)$ time.

We'll define an important concept: *expanded Gabriel discs*

Take 2 points p and q. The *Gabriel disc* of p and q is the disc C_{pq} with diameter pq.

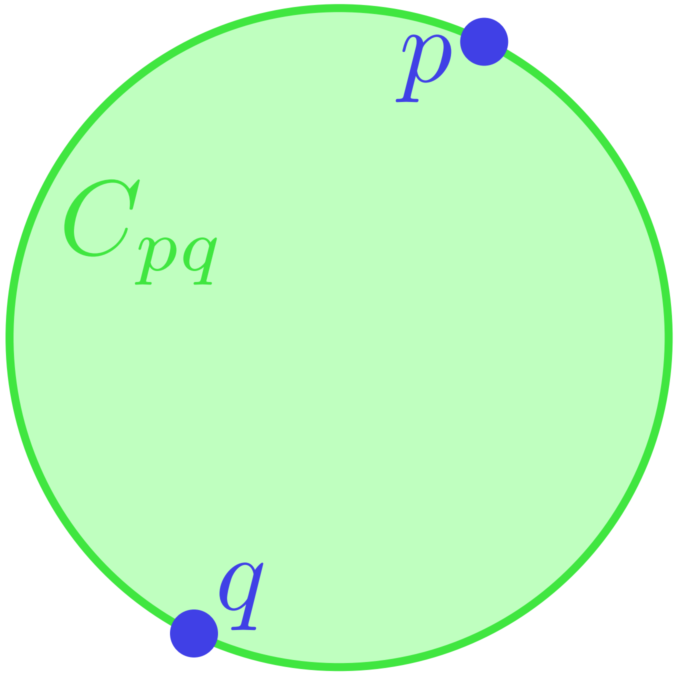
Take 2 points p and q. The *Gabriel disc* of p and q is the disc C_{pq} with diameter pq.

Take 2 points p and q. The *Gabriel disc* of p and q is the disc C_{pq} with diameter pq.



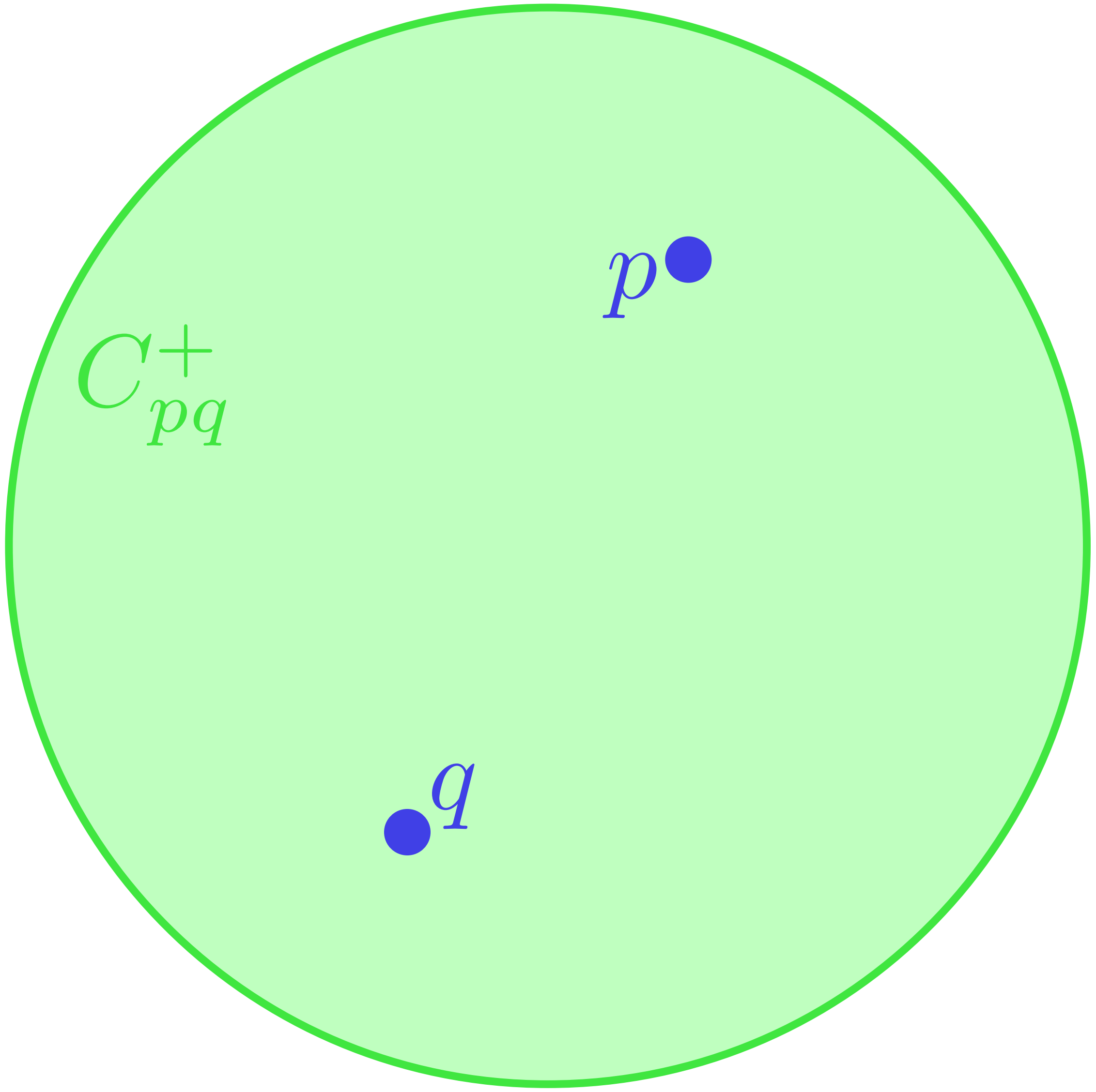
Take 2 points p and q. The *Gabriel disc* of p and q is the disc C_{pq} with diameter pq.

The *expanded Gabriel disc* C_{pq}^{+} is the same, but with radius $+2$.



Take 2 points p and q. The *Gabriel disc* of p and q is the disc C_{pq} with diameter pq.

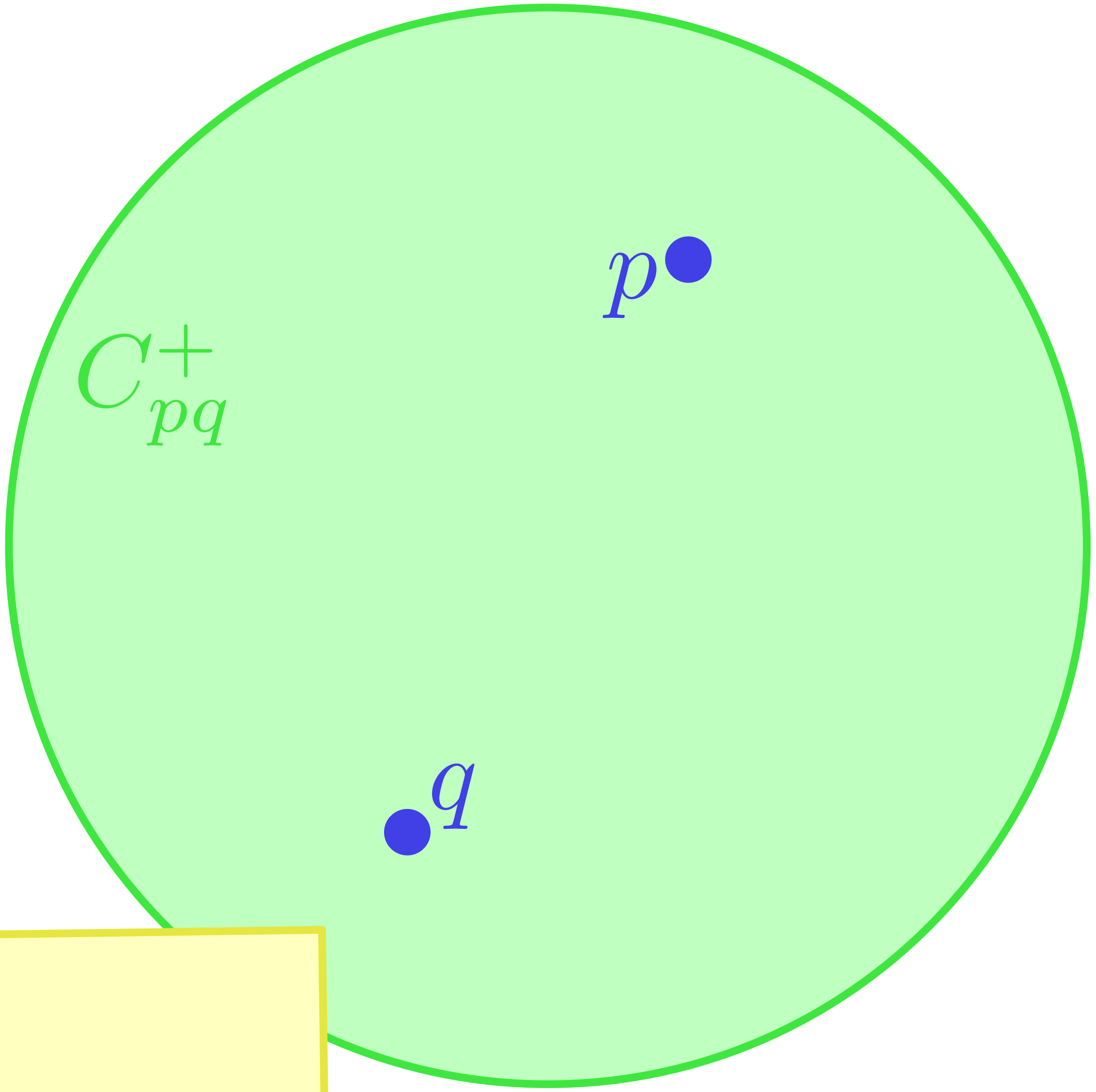
The *expanded Gabriel disc* C_{pq}^+ is the same, but with radius $+2$.



Take 2 points p and q. The *Gabriel disc* of p and q is the disc C_{pq} with diameter pq.

The *expanded Gabriel disc* C^+_{pq} is the same, but with radius +2.

LEMMA
If no point $r \in P$ lies inside C^+_{pq}, then $\hat{p}\hat{q}$ is an edge of $\hat{T}$.

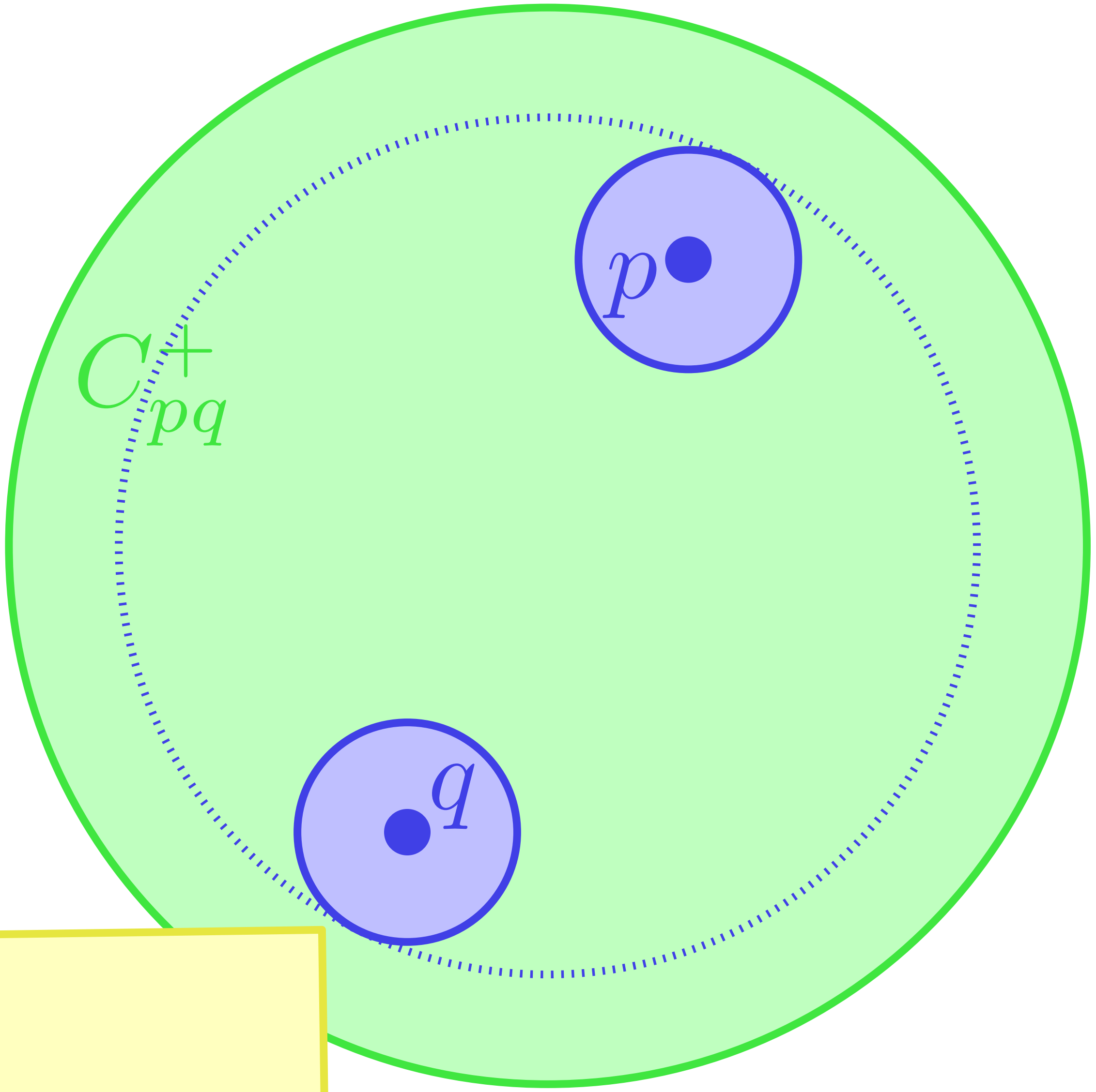


Take 2 points p and q. The *Gabriel disc* of p and q is the disc C_{pq} with diameter pq.

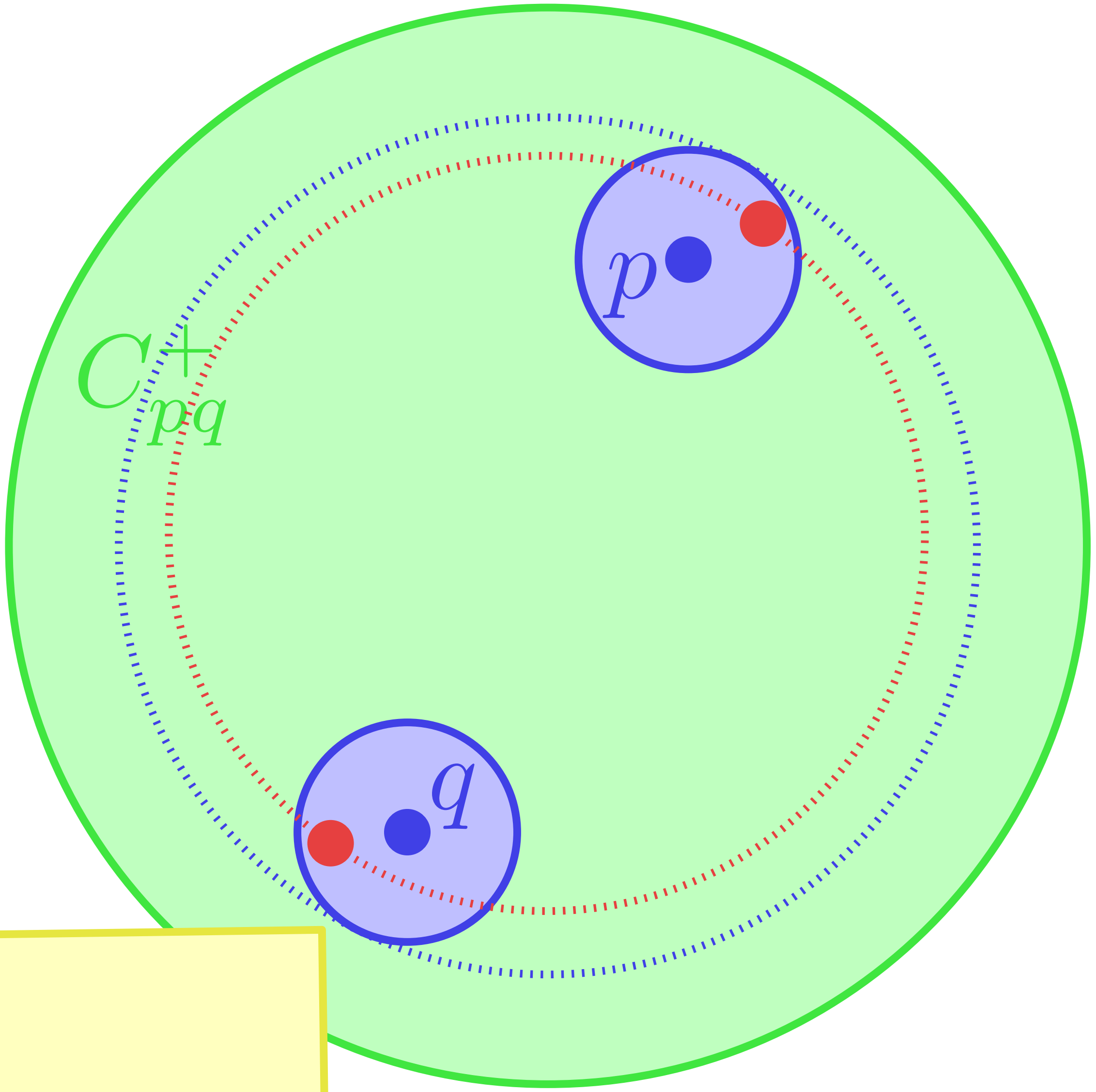
The *expanded Gabriel disc* C^+_{pq} is the same, but with radius $+2$.

LEMMA

If no point $r \in P$ lies inside C^+_{pq}, then $\hat{p}\hat{q}$ is an edge of $\hat{T}$.



Take 2 points p and q. The Gabriel disc of p and q is the disc C_{pq} with diameter pq.



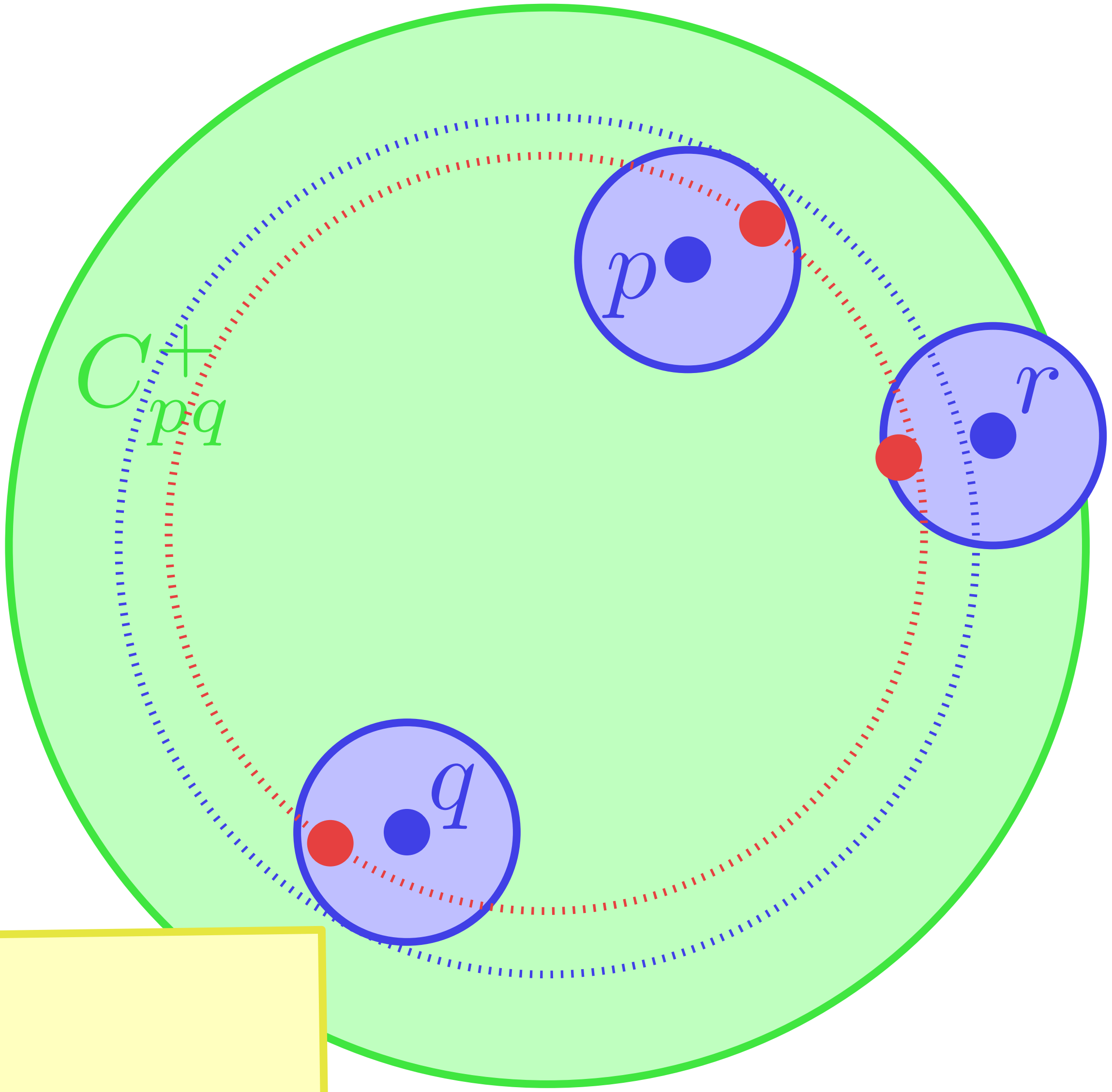
The expanded Gabriel disc C^+_{pq} is the same, but with radius $+2$.

LEMMA
If no point $r \in P$ lies inside C^+_{pq}, then $\hat{p}\hat{q}$ is an edge of $\hat{T}$.

Take 2 points p and q. The *Gabriel disc* of p and q is the disc C_{pq} with diameter pq.

The expanded *Gabriel disc* C_{pq}^+ is the same, but with radius $+2$.

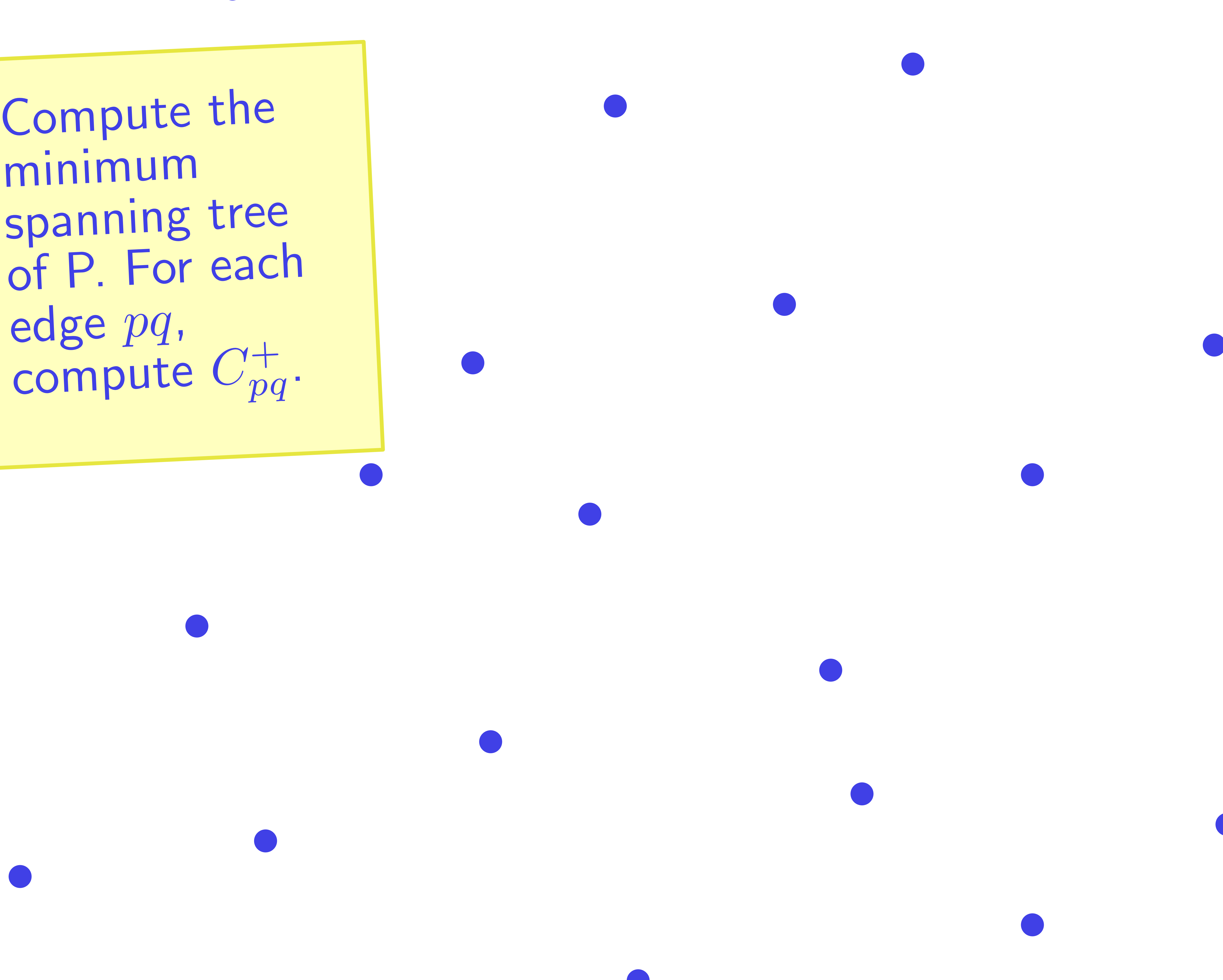
LEMMA
If no point $r \in P$ lies inside C_{pq}^+, then $\hat{p}\hat{q}$ is an edge of $\hat{T}$.

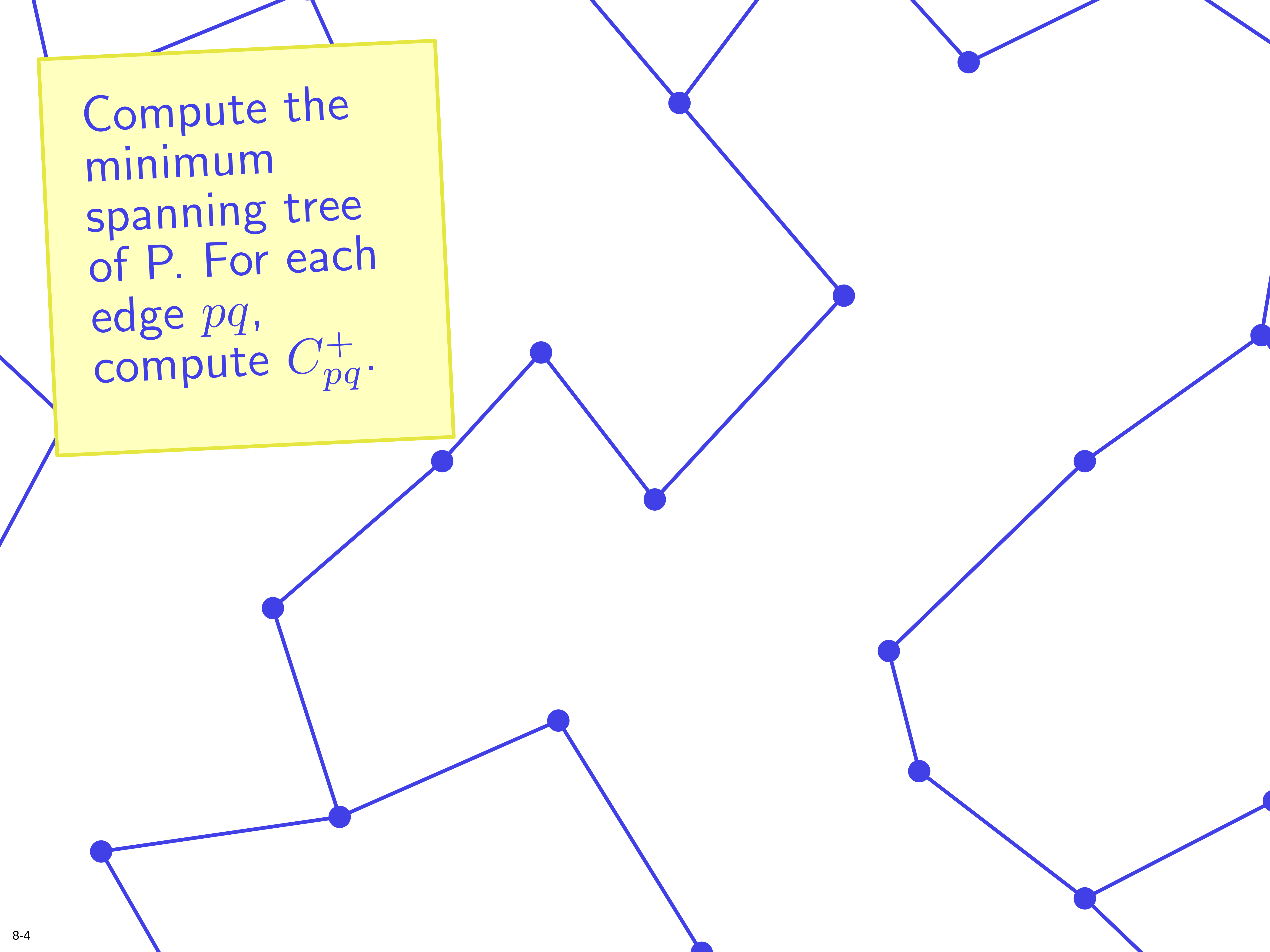


Preprocessing: we're going to compute a *minimum spanning tree*!

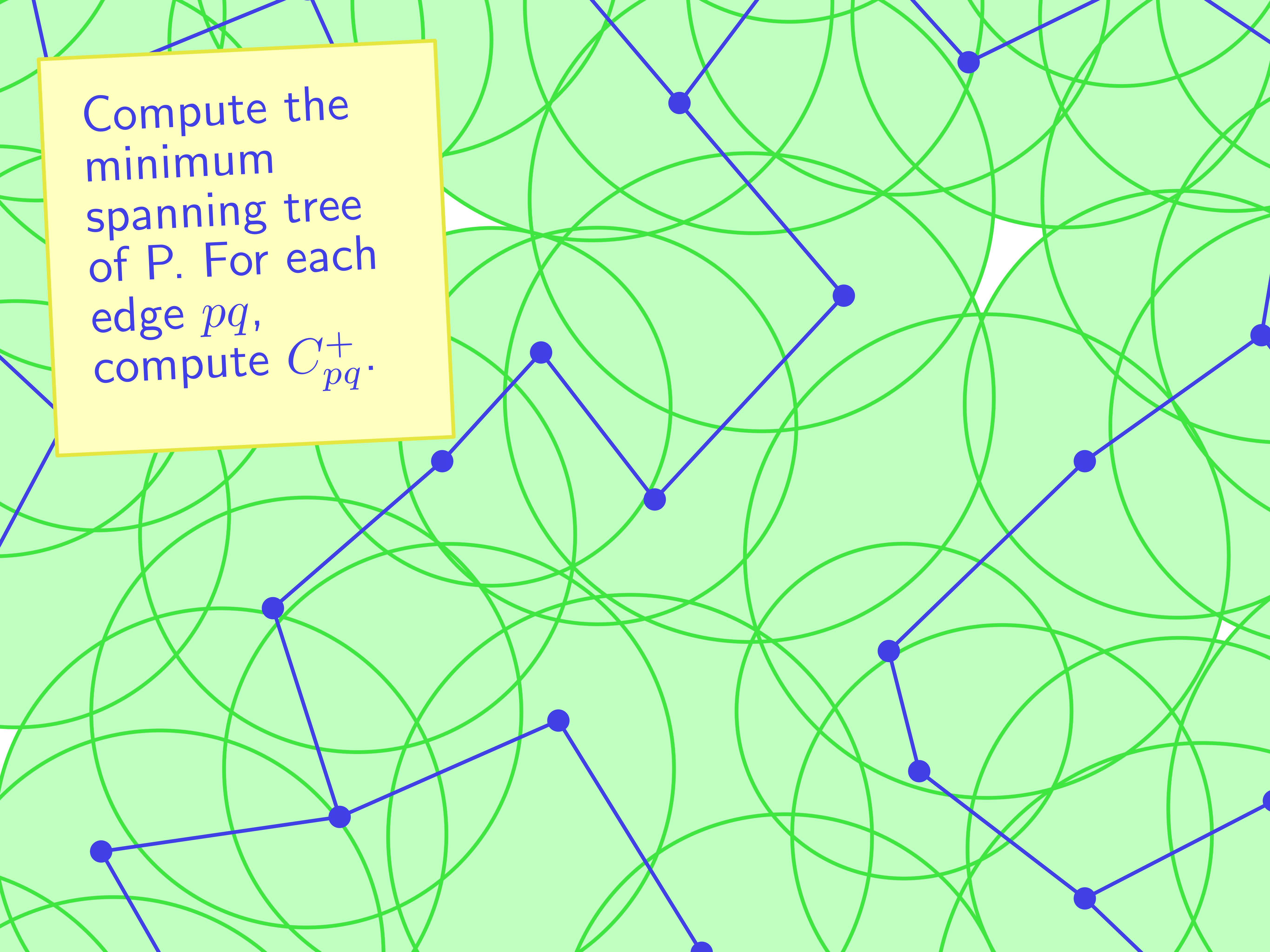
Compute the minimum spanning tree of P. For each edge pq, compute C^+_{pq}.

Compute the minimum spanning tree of P. For each edge pq, compute C^+_{pq}.

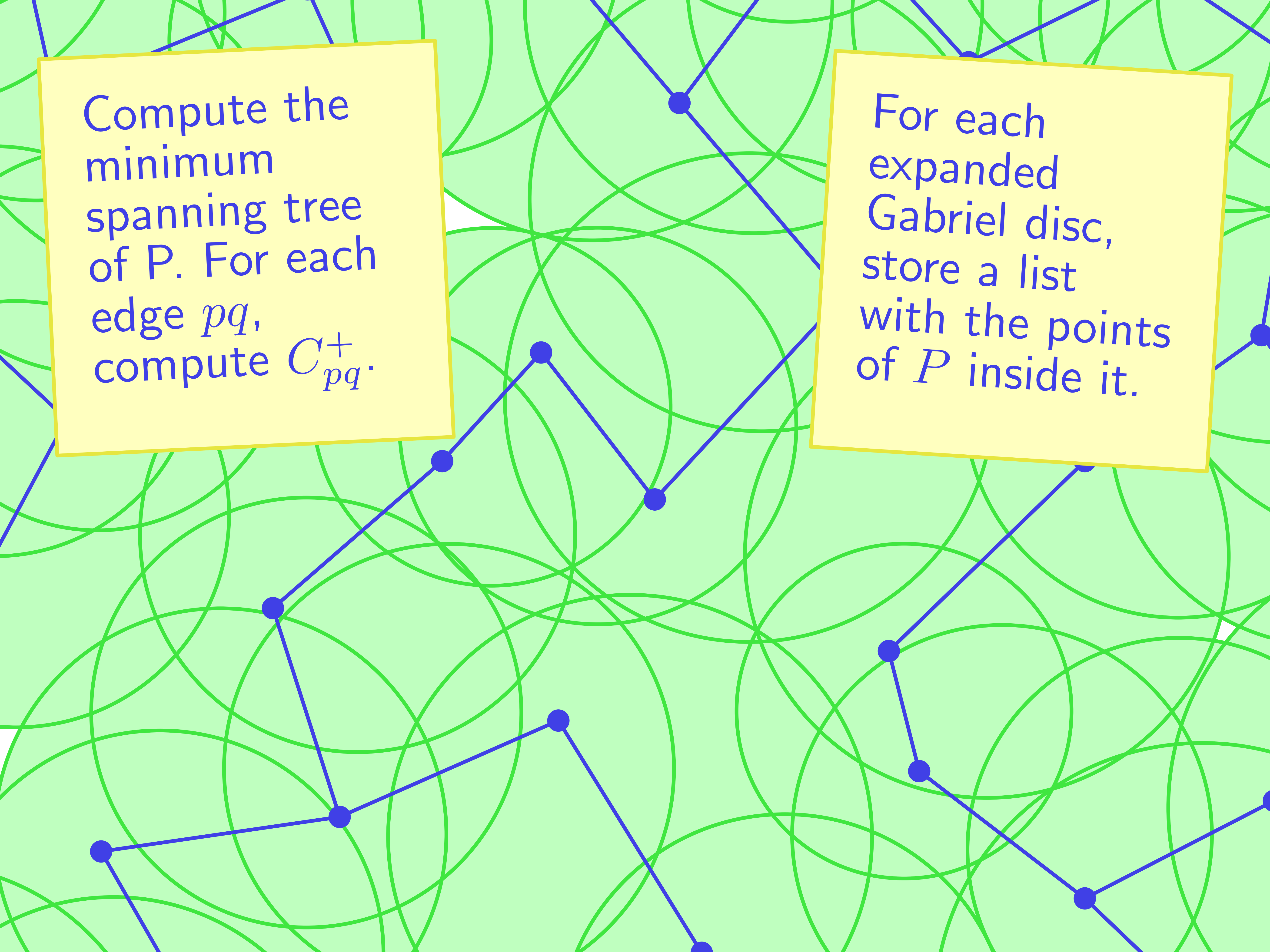




Compute the minimum spanning tree of P. For each edge pq, compute C^+_{pq}.



Compute the minimum spanning tree of P. For each edge pq, compute C_{pq}^{+}.


Compute the minimum spanning tree of P. For each edge pq, compute C_{pq}^{+}.
For each expanded Gabriel disc, store a list with the points of P inside it.

Compute the minimum spanning tree of P. For each edge pq, compute C_{pq}^{+}.

For each expanded Gabriel disc, store a list with the points of P inside it.

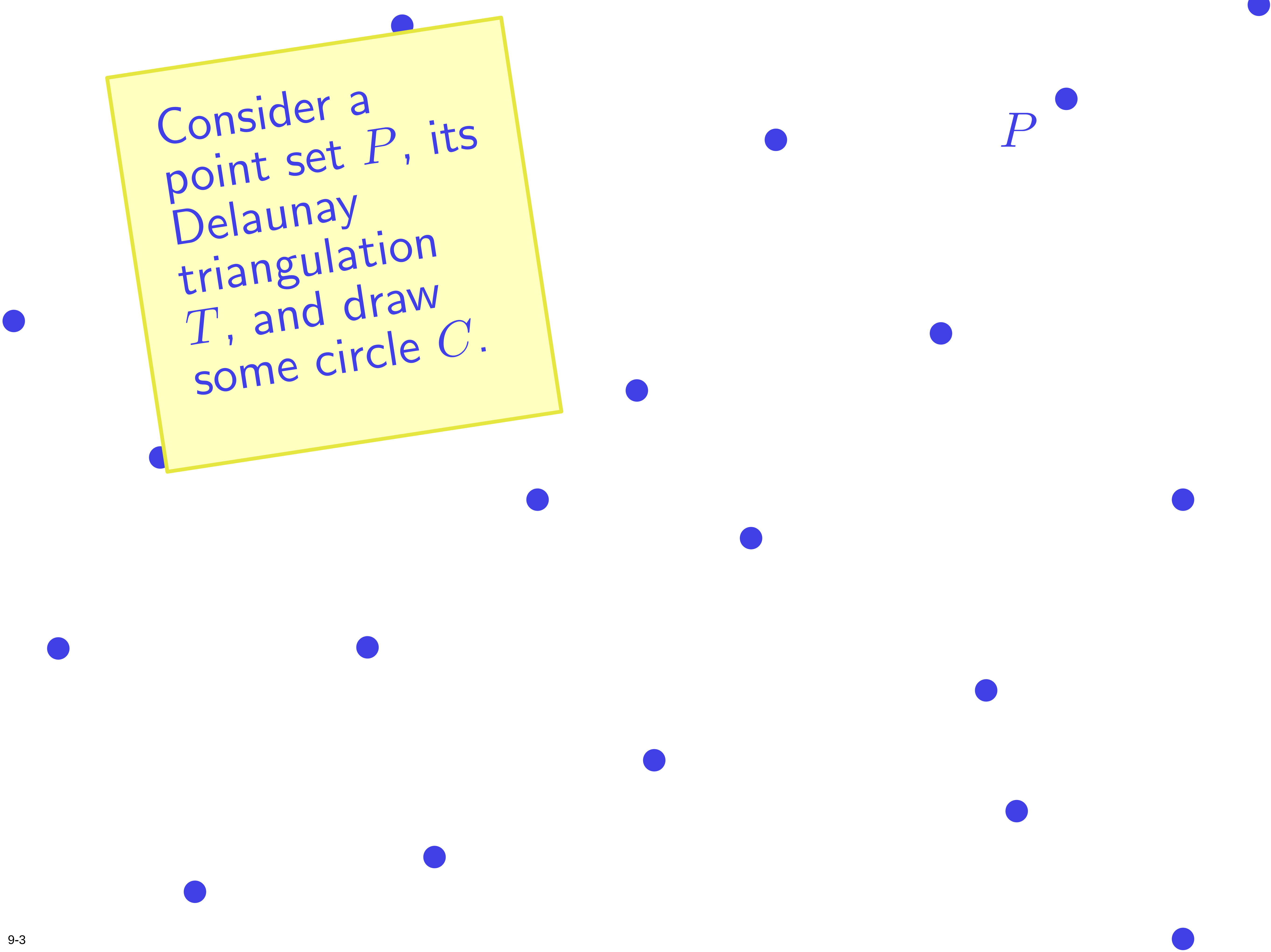
LEMMA

The total complexity of these lists is $O(n)$.

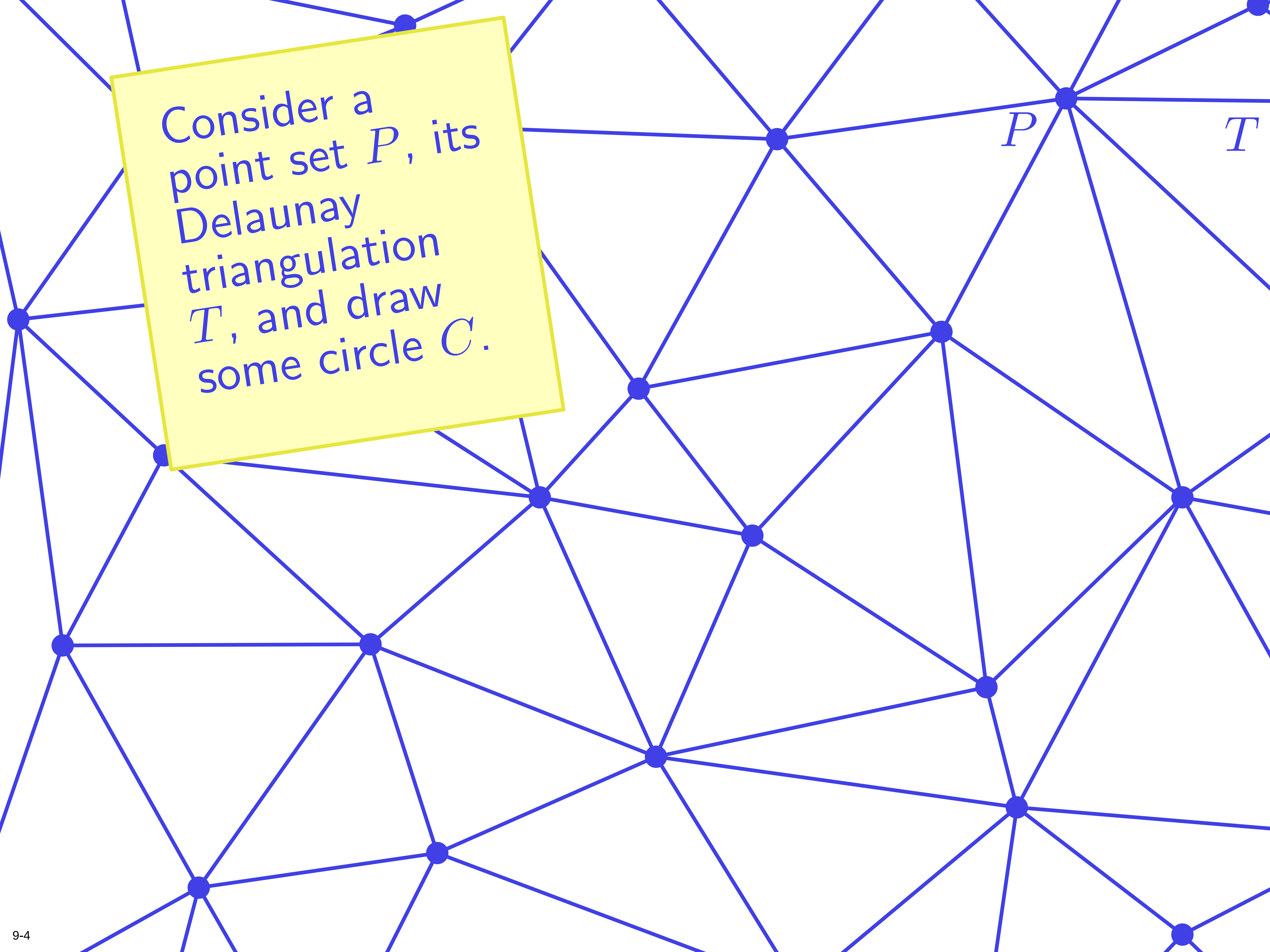
Let's do a nice small technical lemma.

Consider a point set P, its Delaunay triangulation T, and draw some circle C.

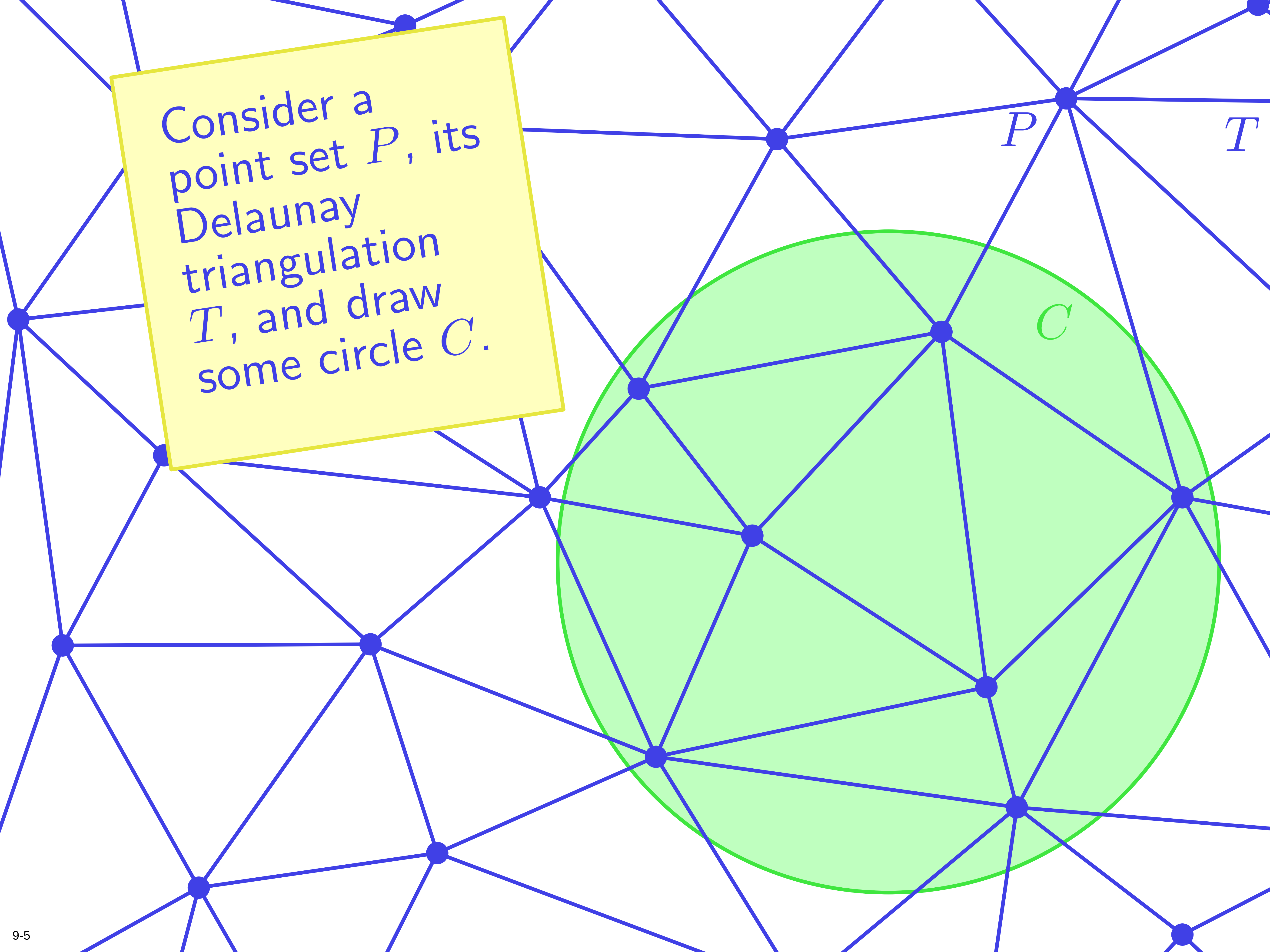
Consider a point set P, its Delaunay triangulation T, and draw some circle C.

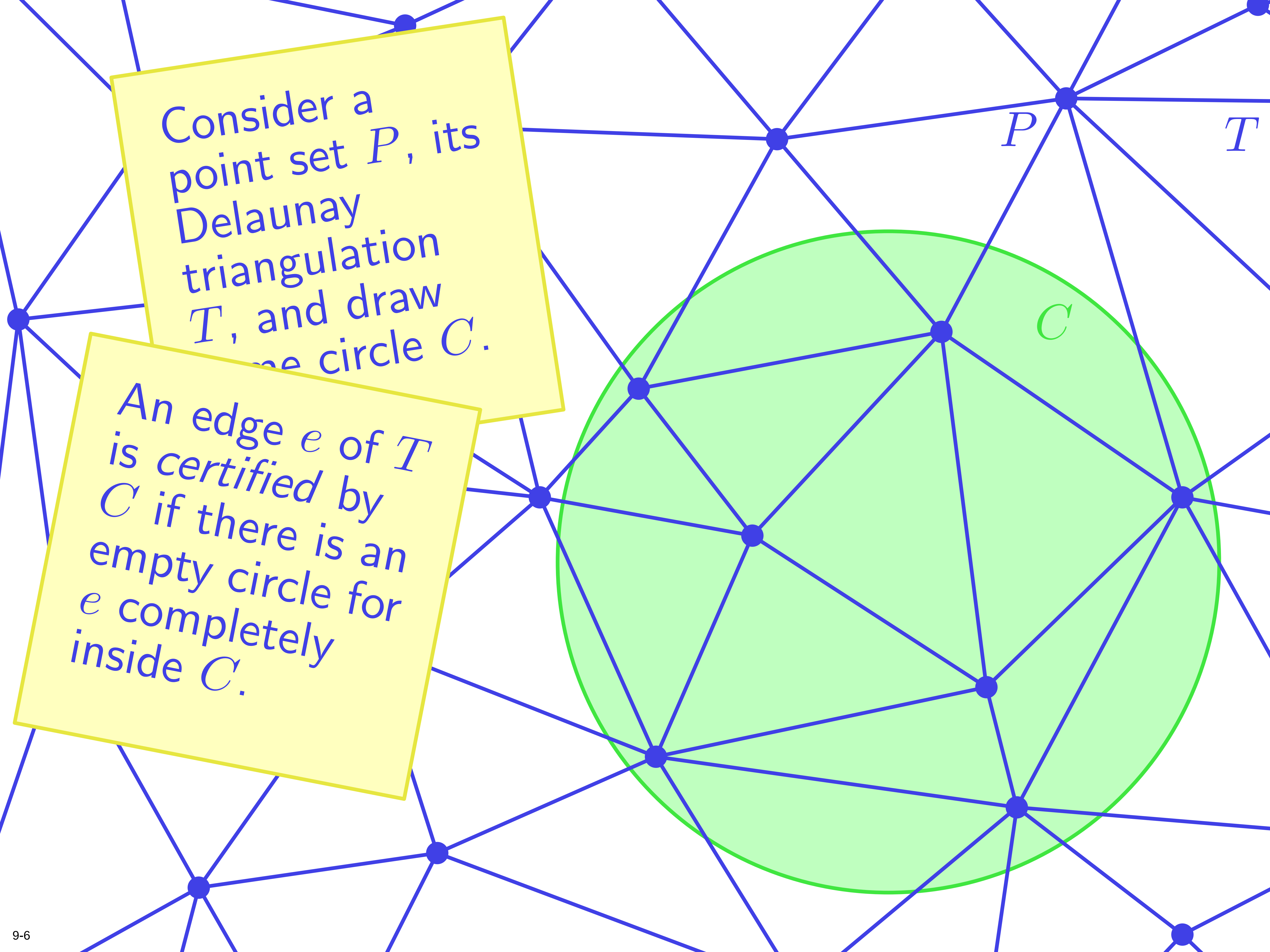


Consider a point set P, its Delaunay triangulation T, and draw some circle C.



Consider a point set P, its Delaunay triangulation T, and draw some circle C.



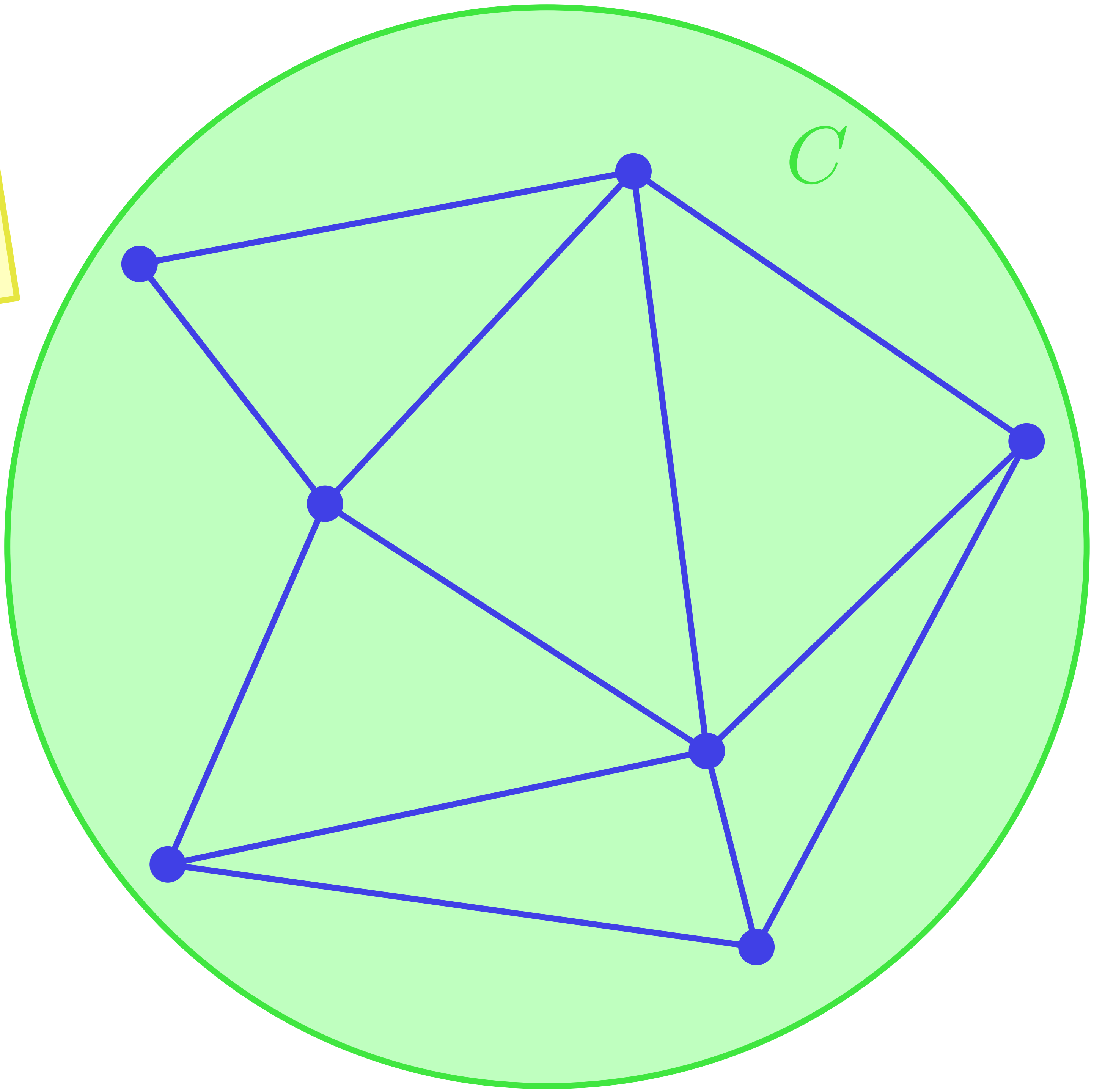


Consider a point set P, its Delaunay triangulation T, and draw me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

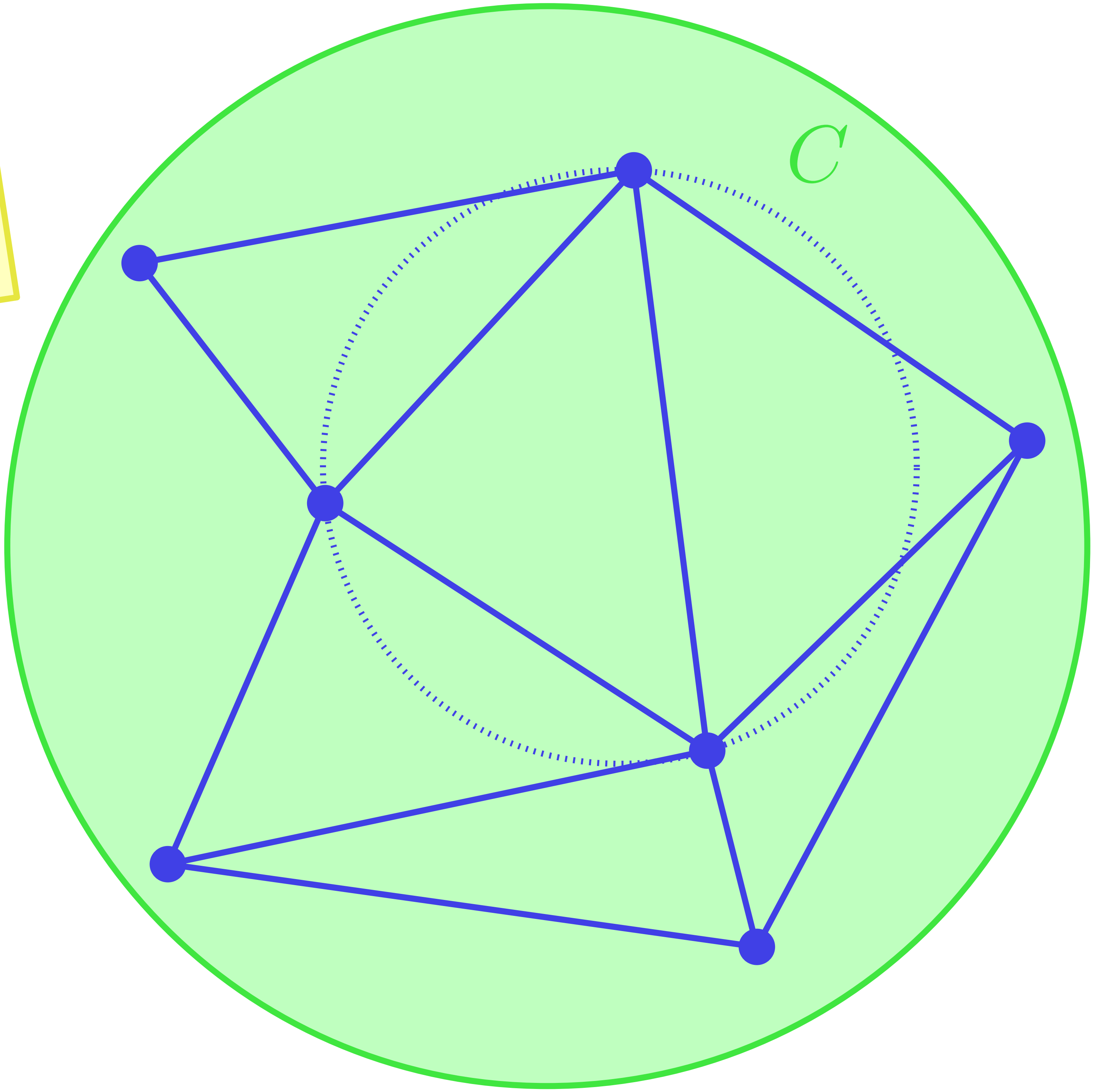
Consider a point set P, its Delaunay triangulation T, and draw some circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.



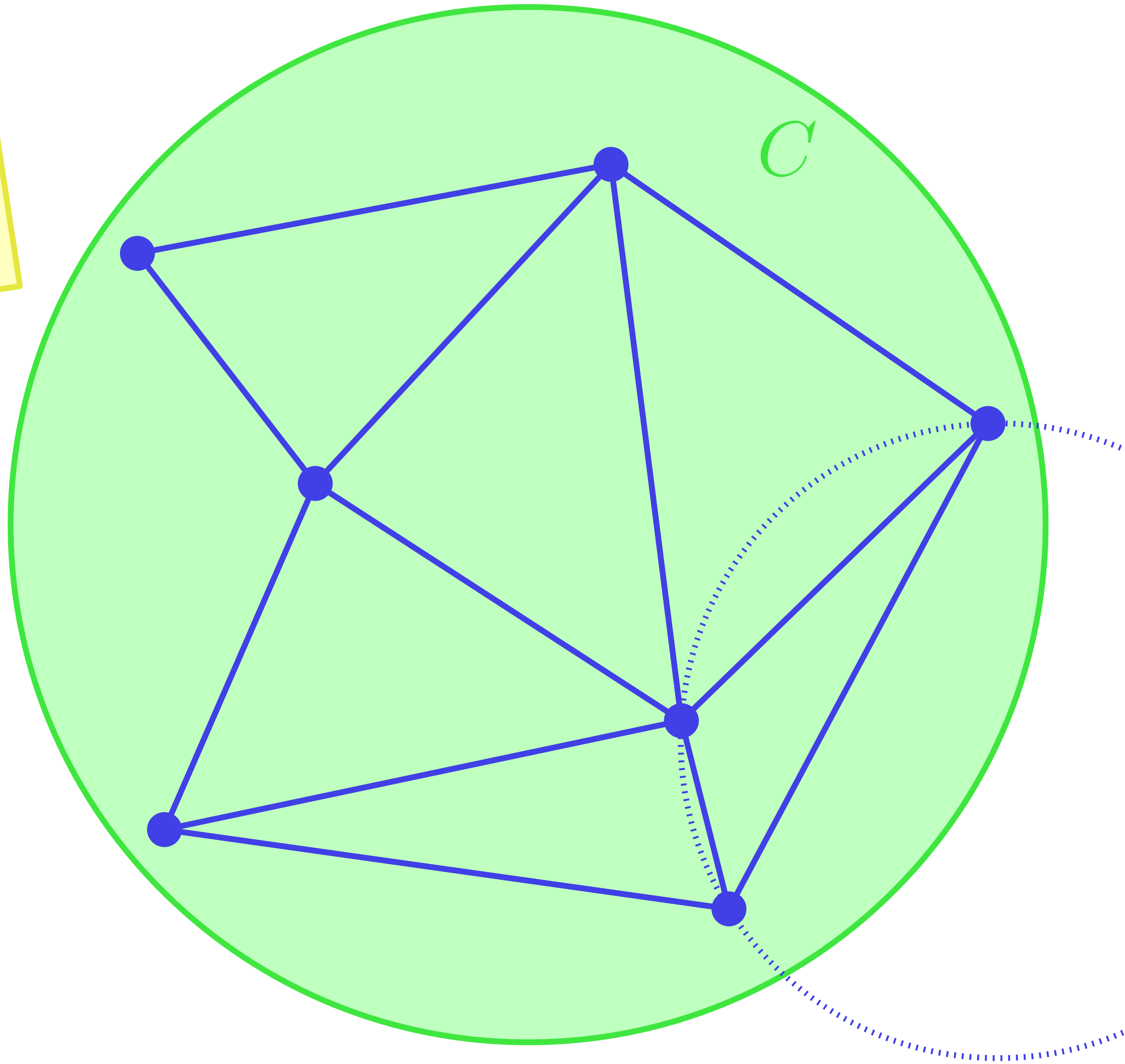
Consider a point set P, its Delaunay triangulation T, and draw me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.



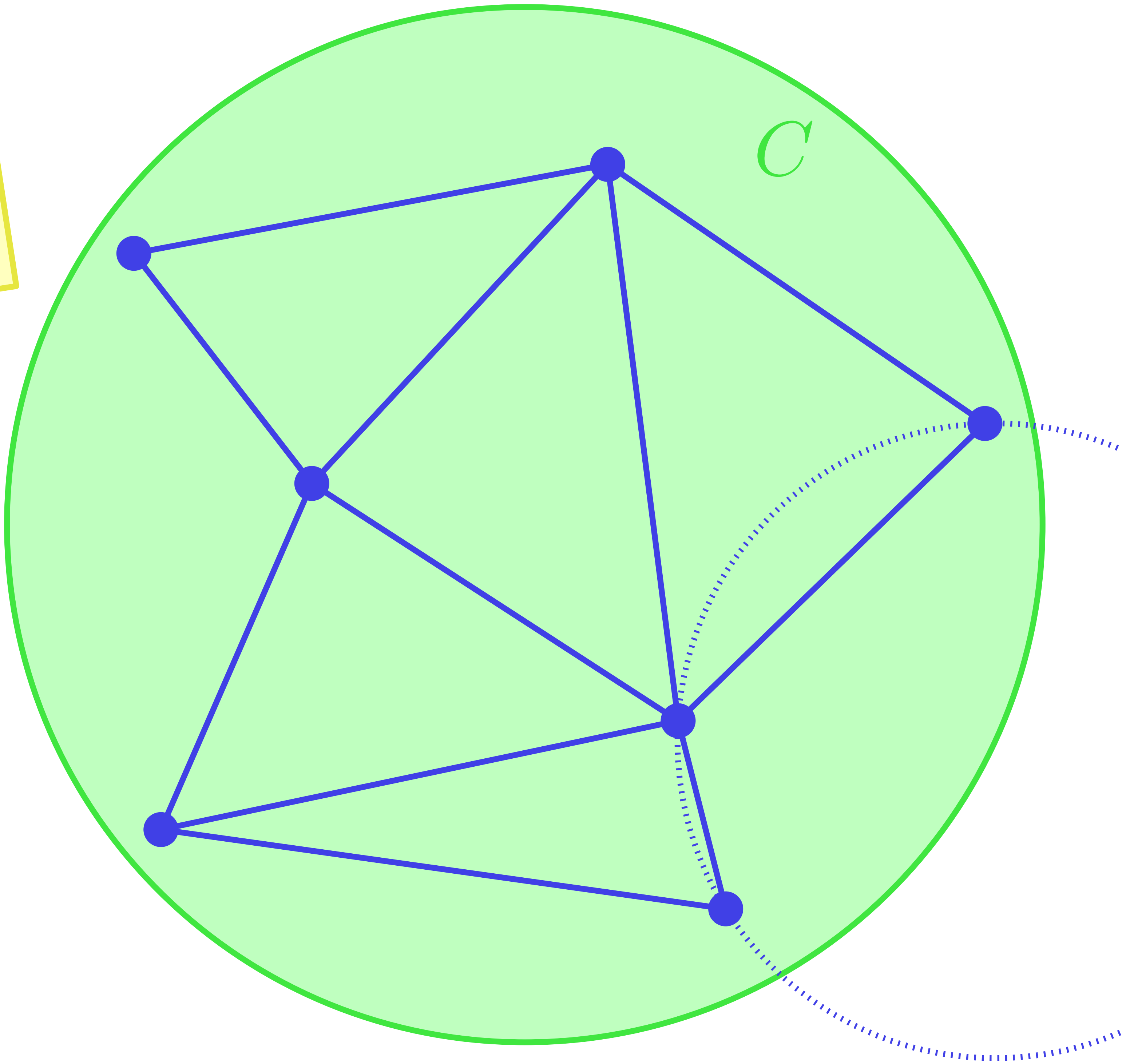
Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.



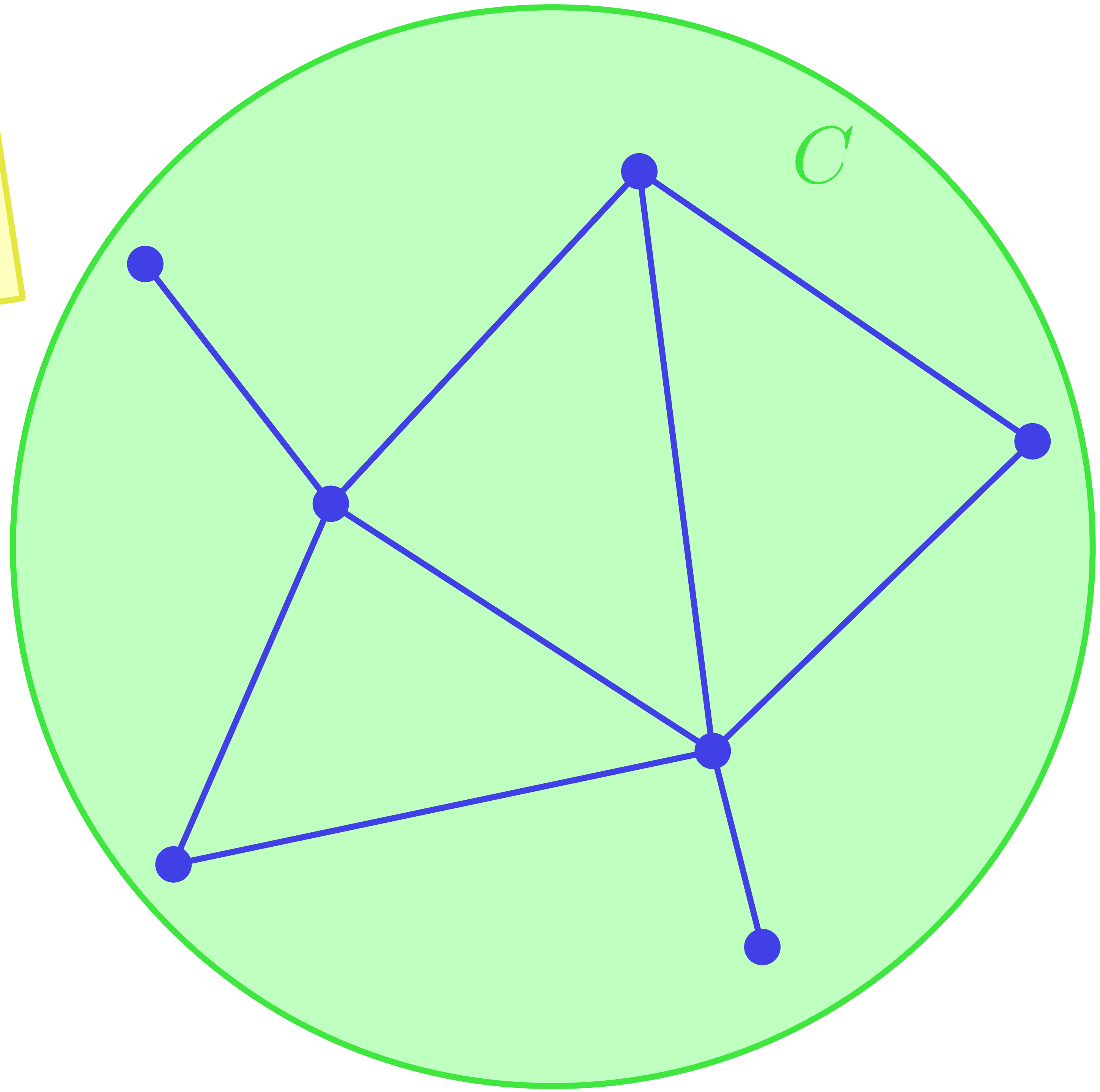
Consider a point set P, its Delaunay triangulation T, and draw me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

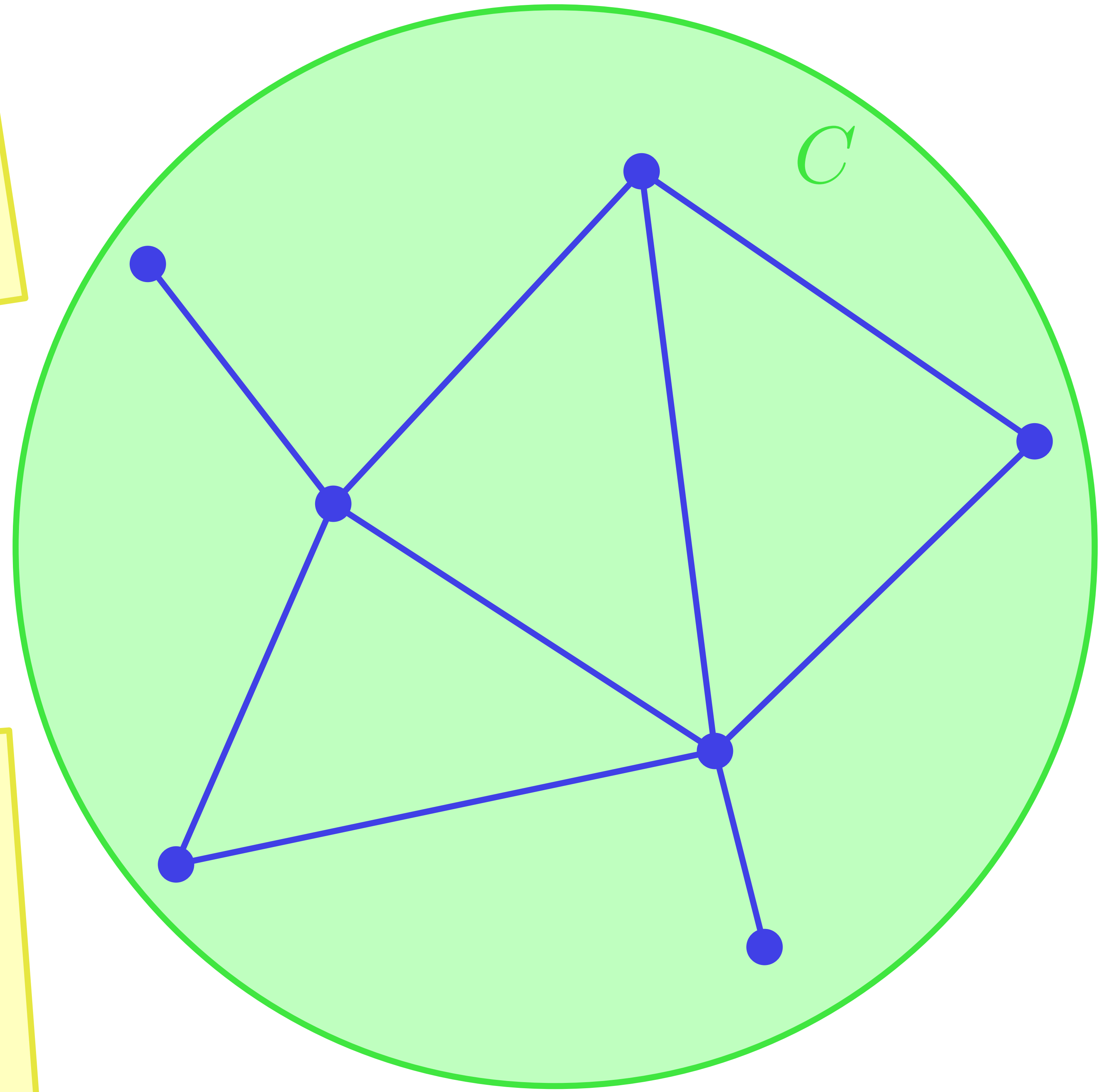
An edge e of T is *certified* by C if there is an empty circle for e completely inside C.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.

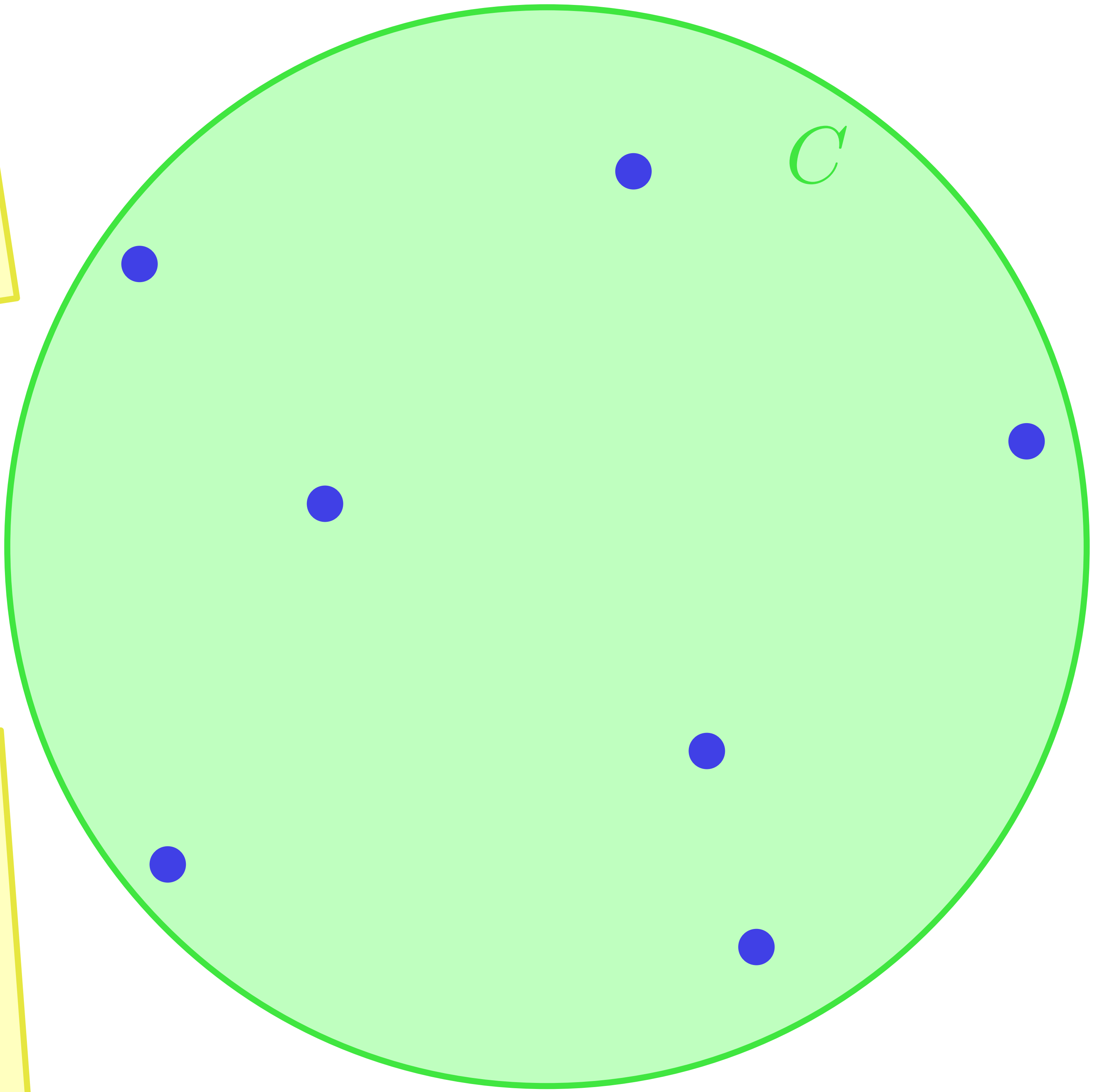
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw [...]me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

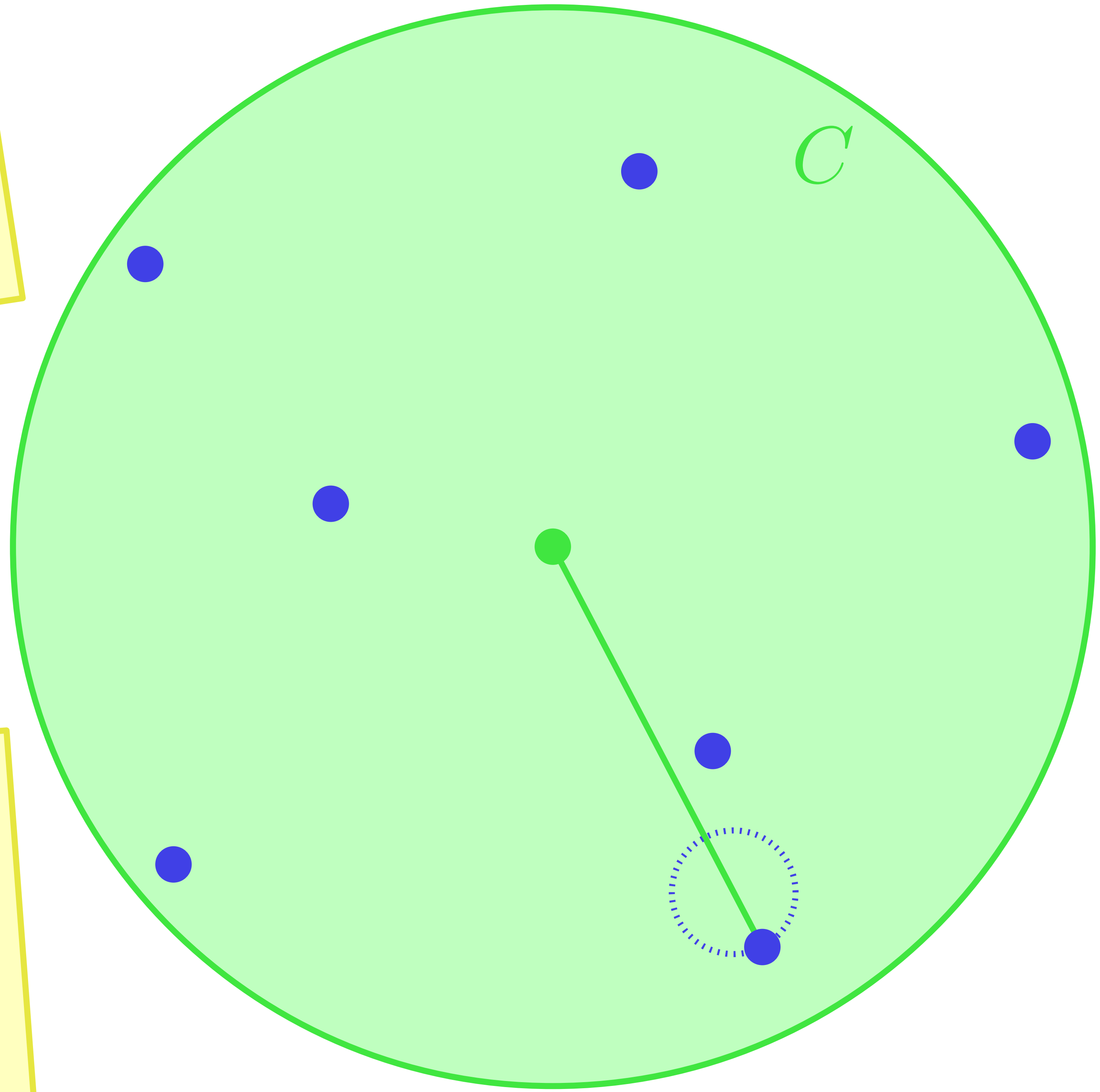
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C

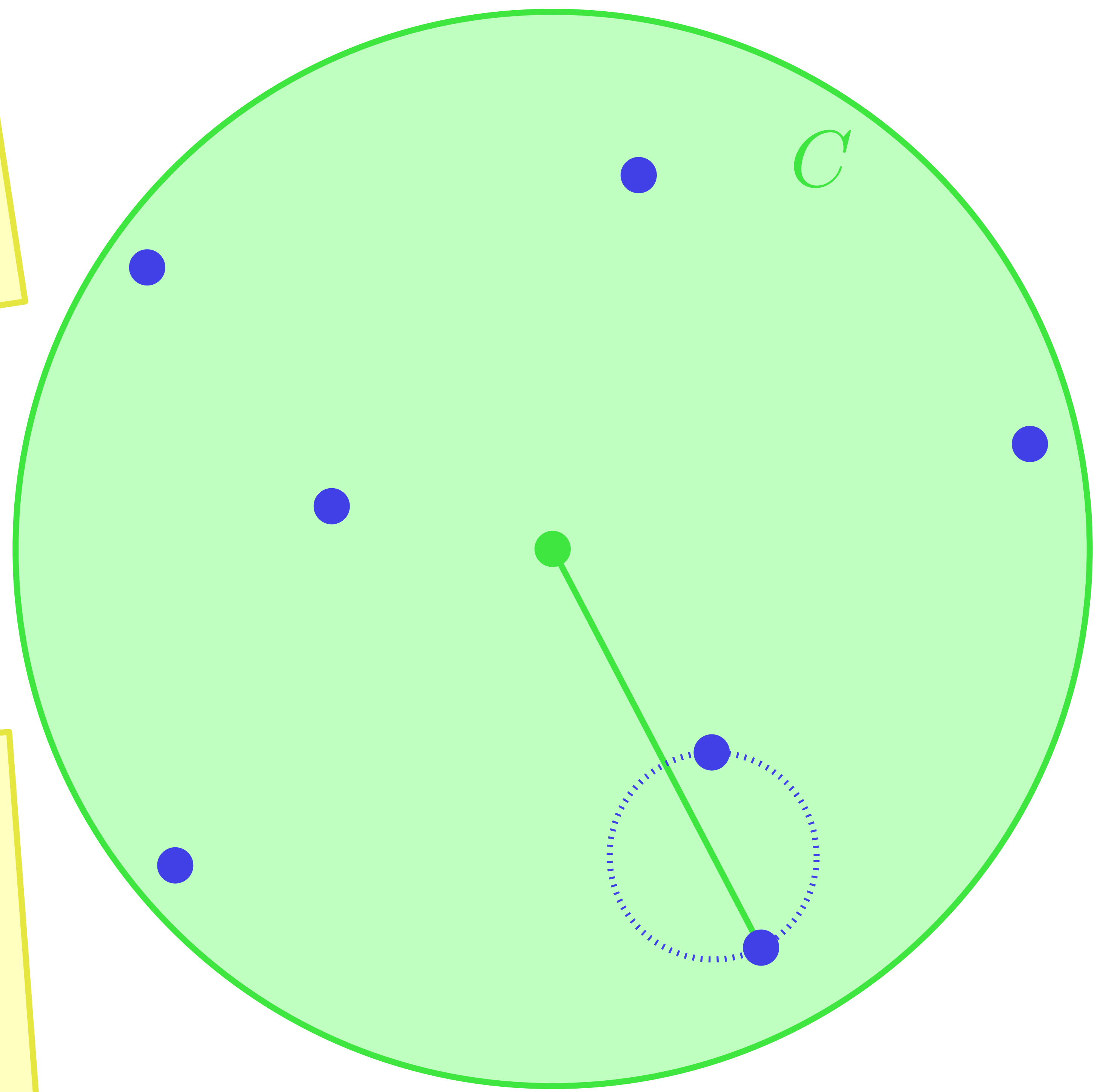
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw
me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

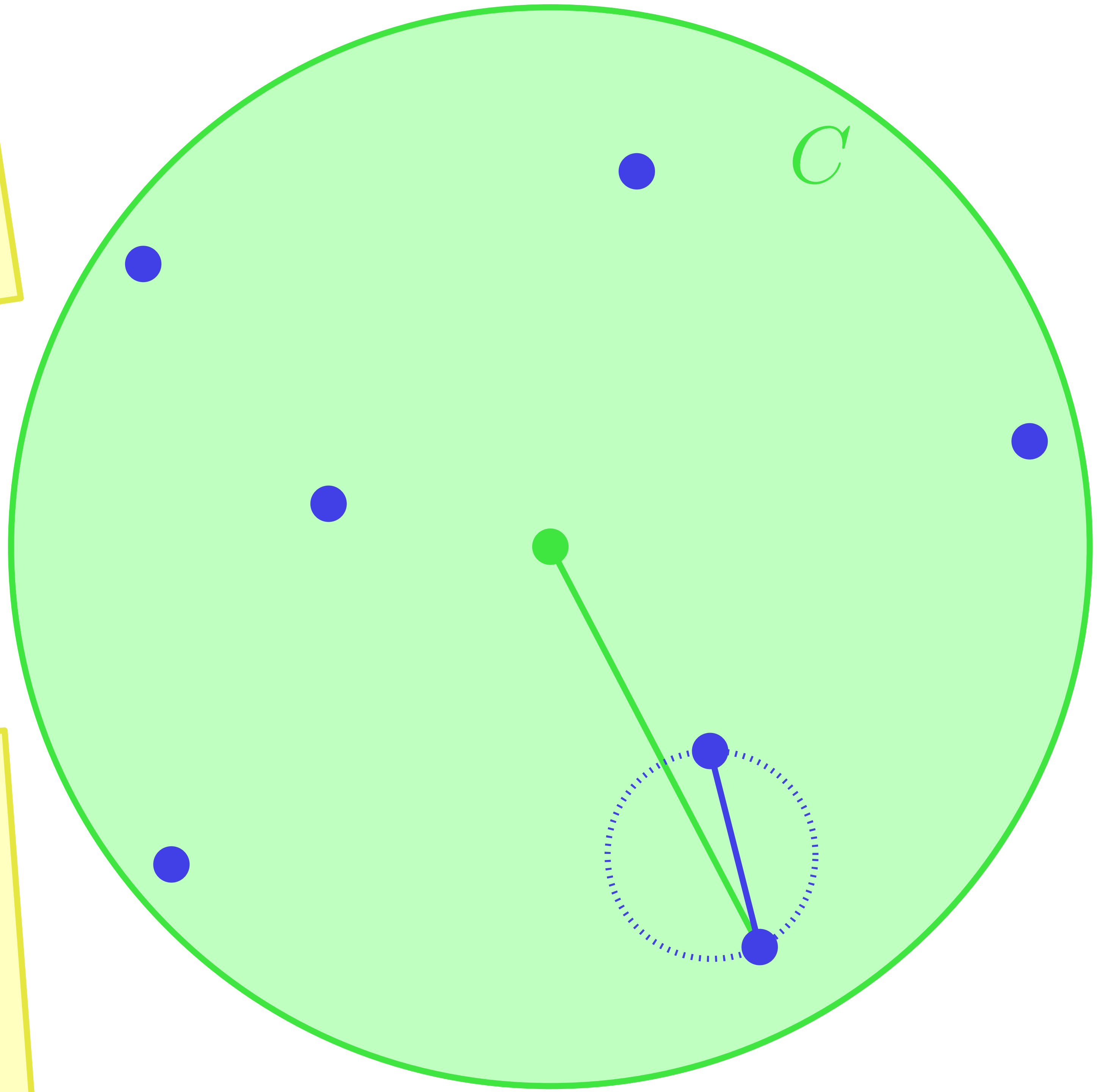
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw
me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

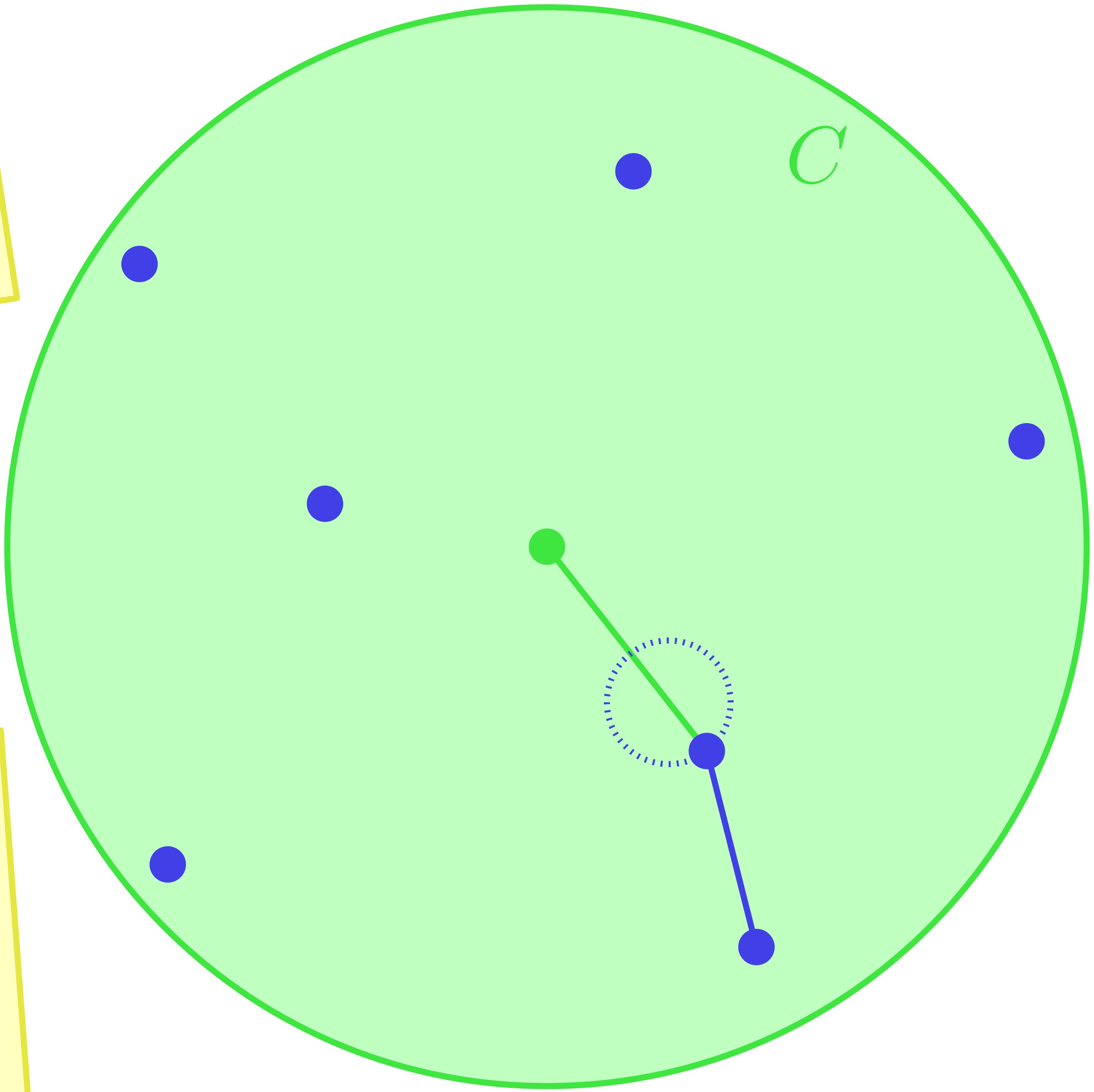
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

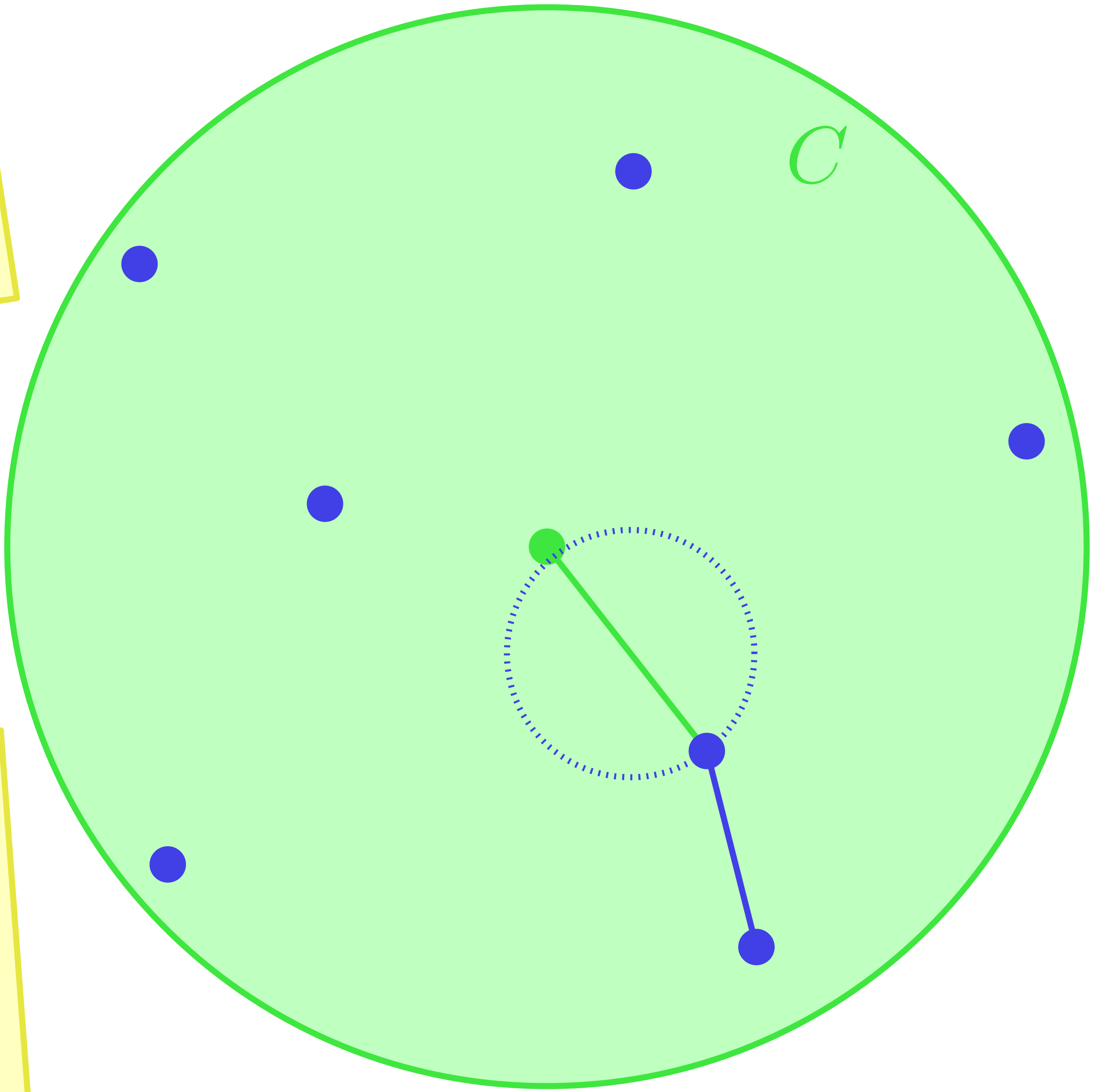
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

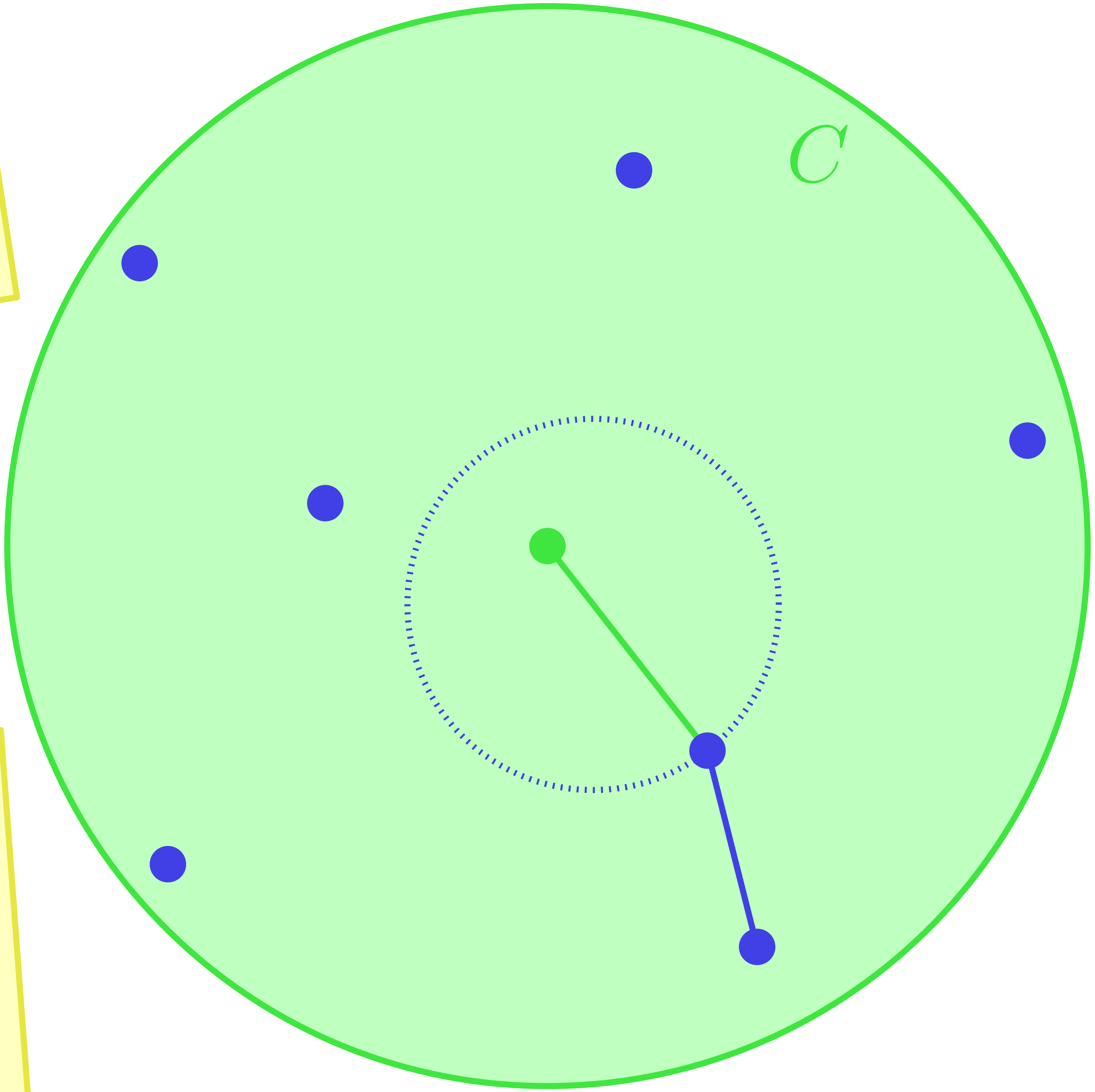
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

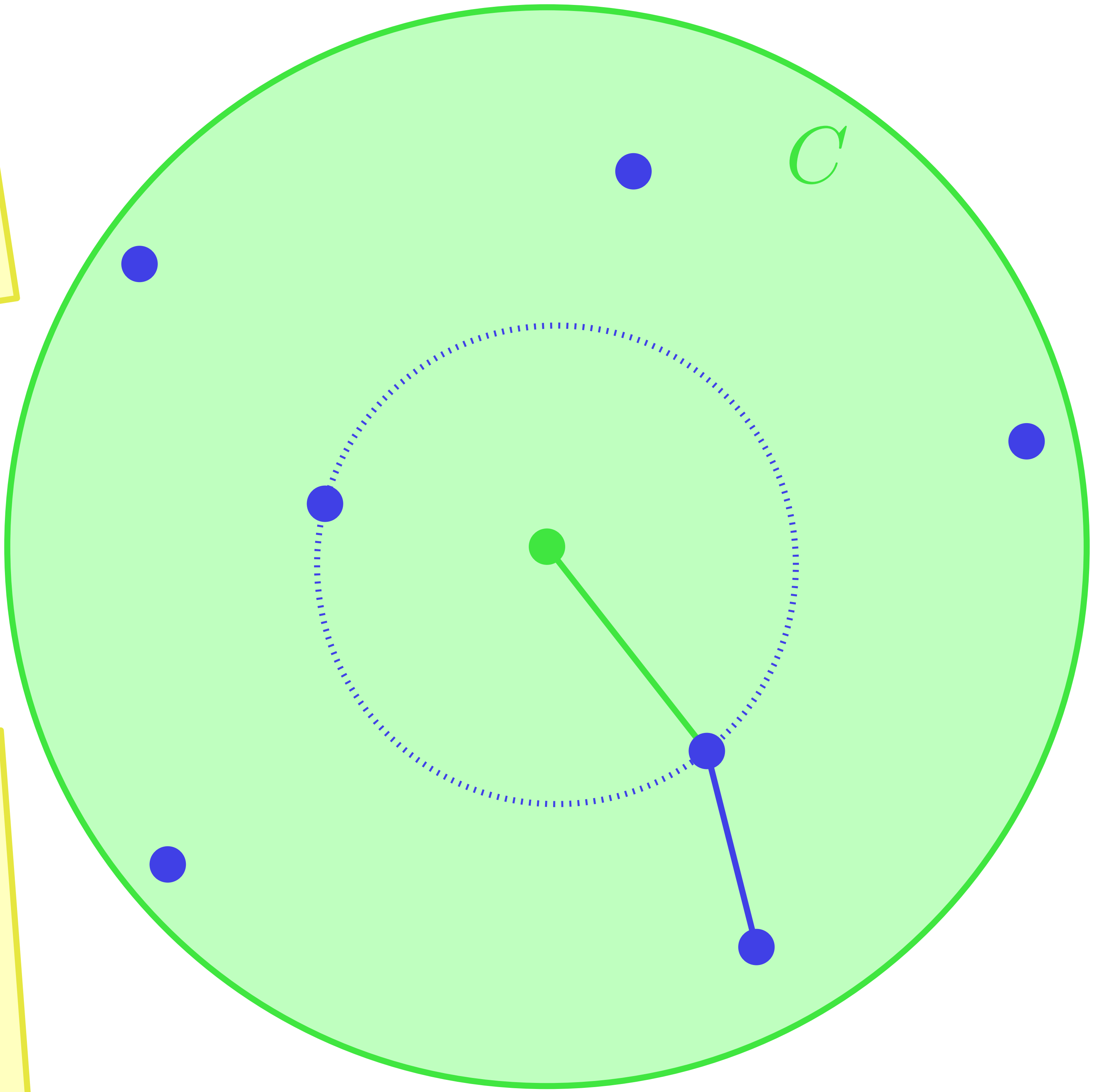
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.

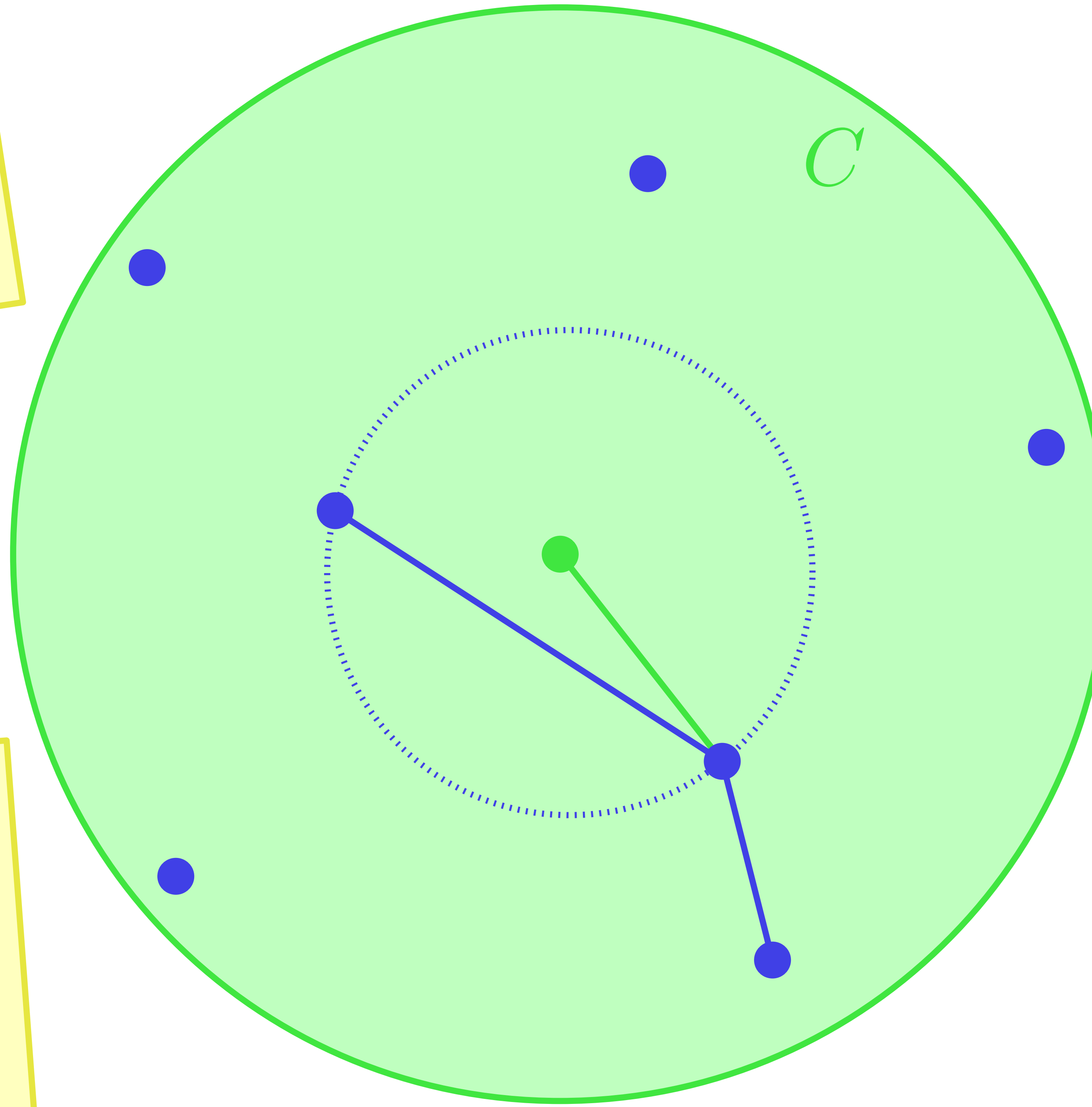
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.

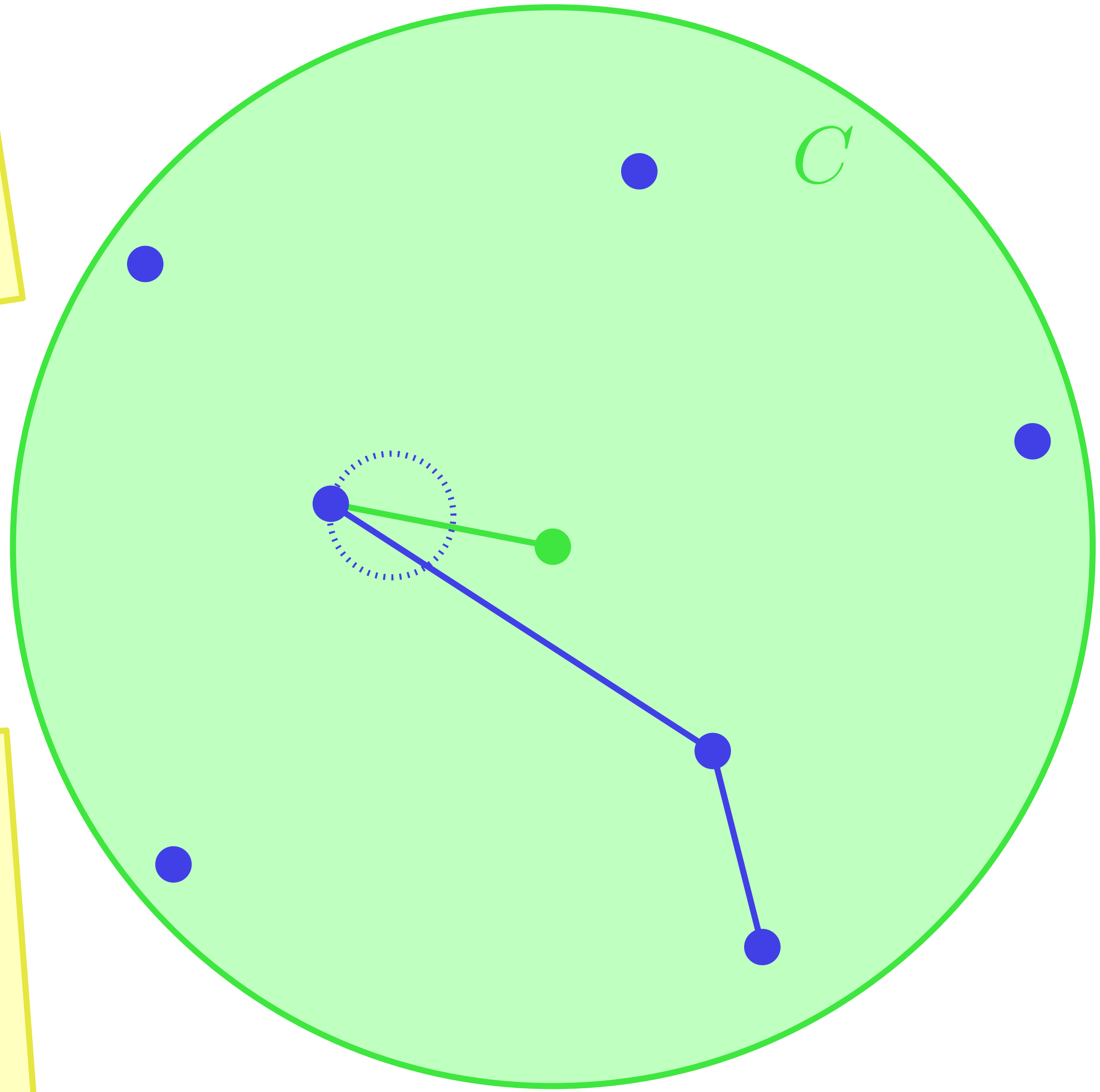
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.

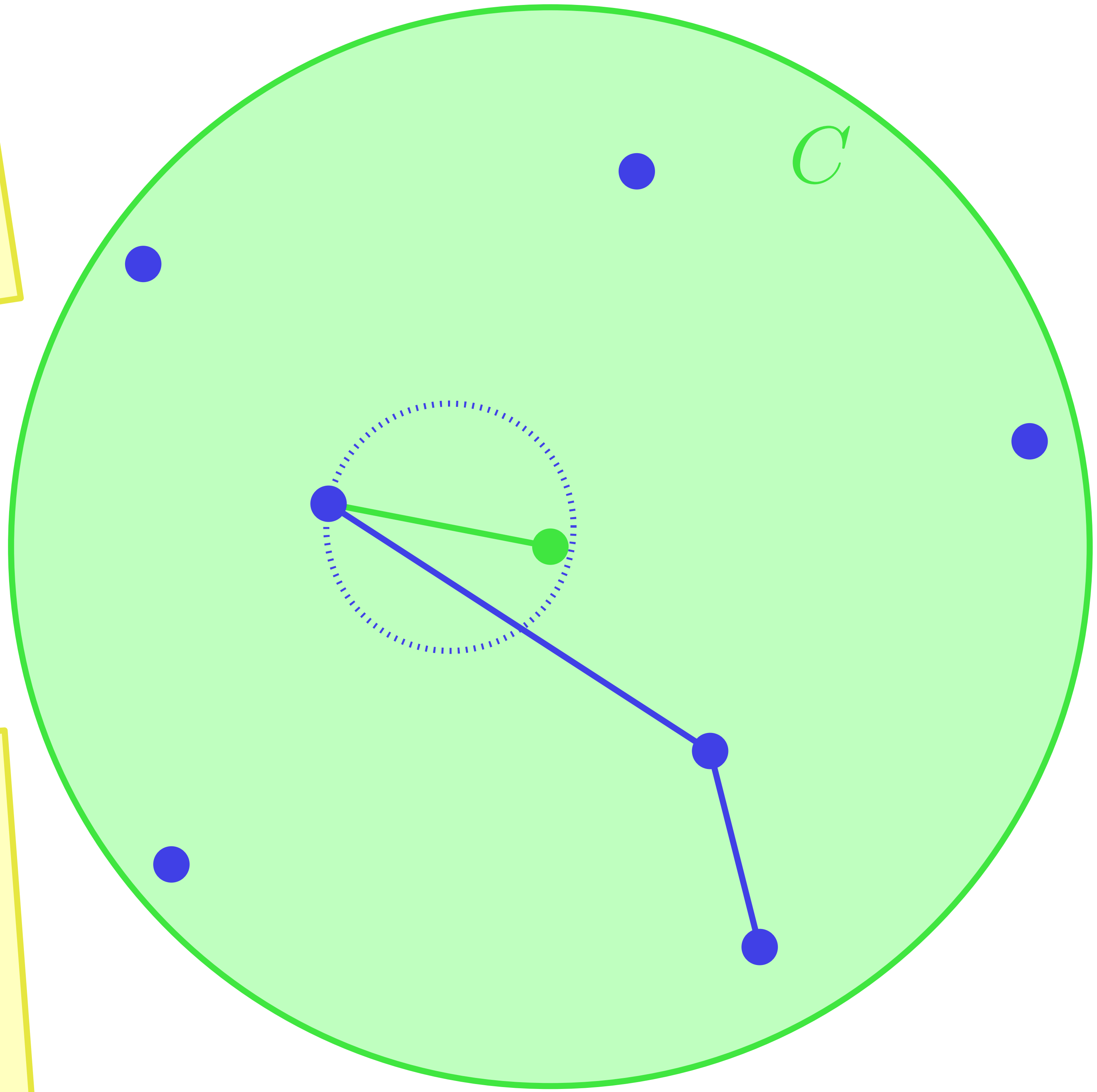
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw [occluded] me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.

LEMMA
The resulting graph is always connected.

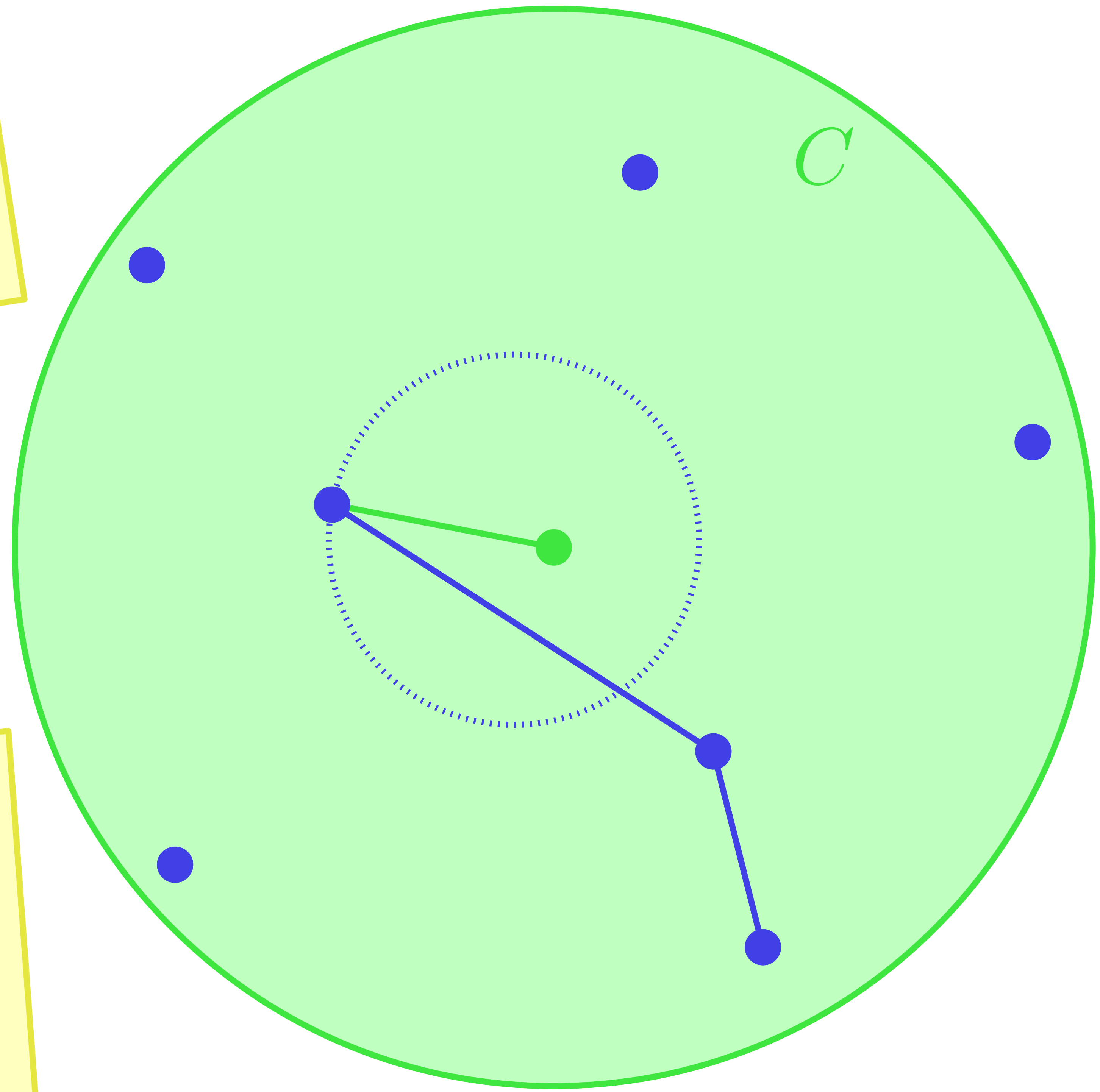


Consider a point set P, its Delaunay triangulation T, and draw
me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.

LEMMA

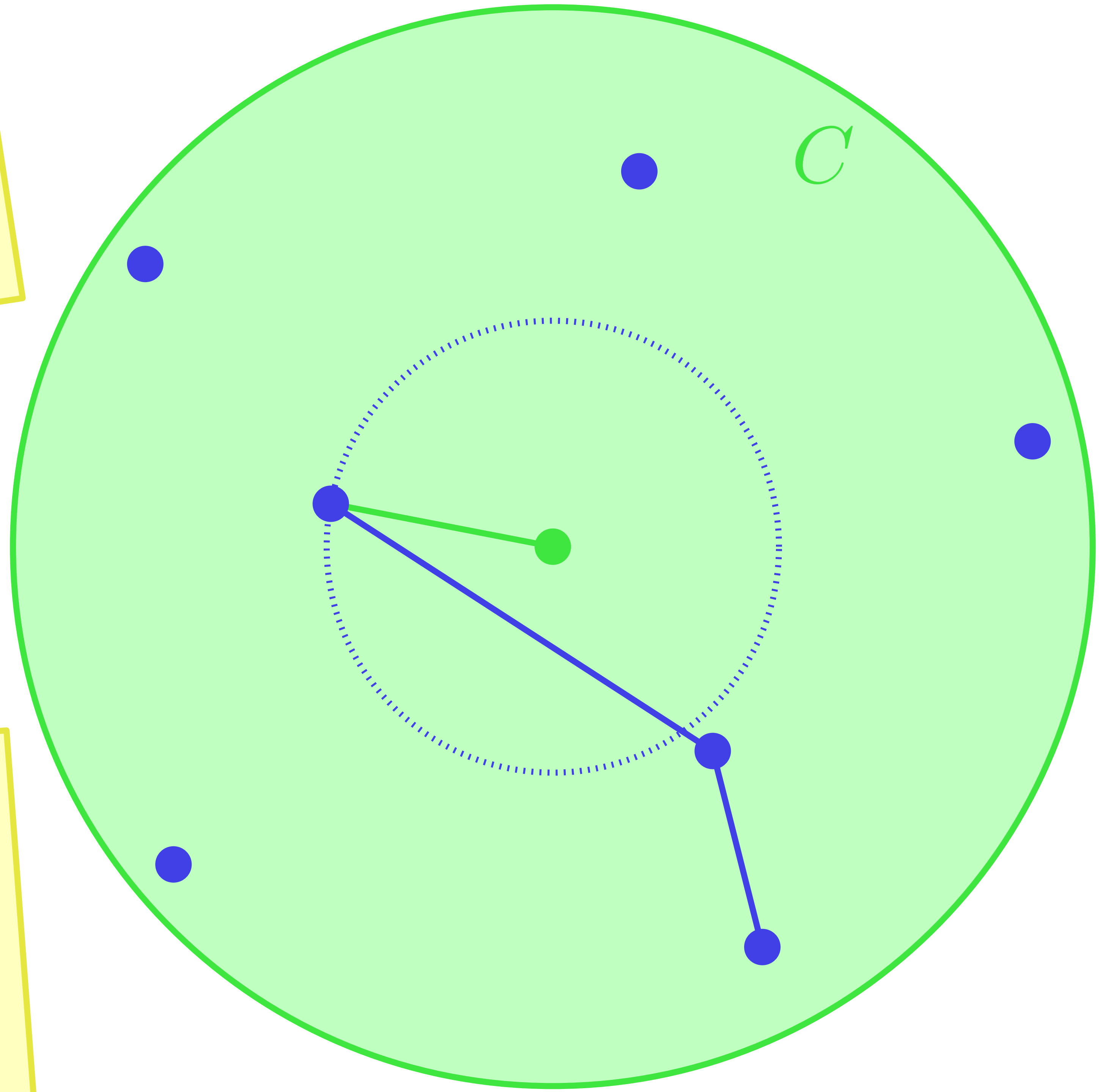
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw …me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.

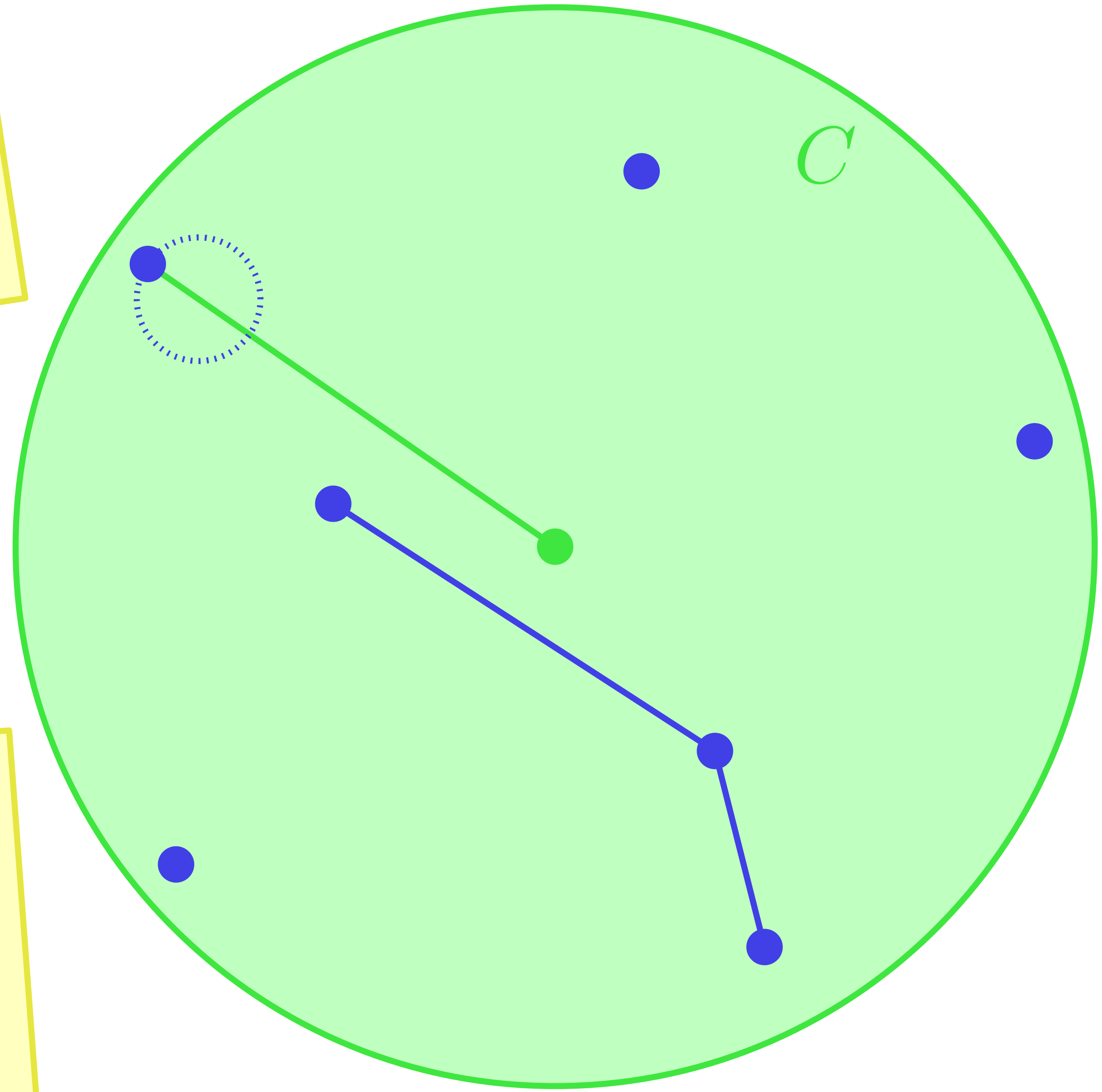
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.

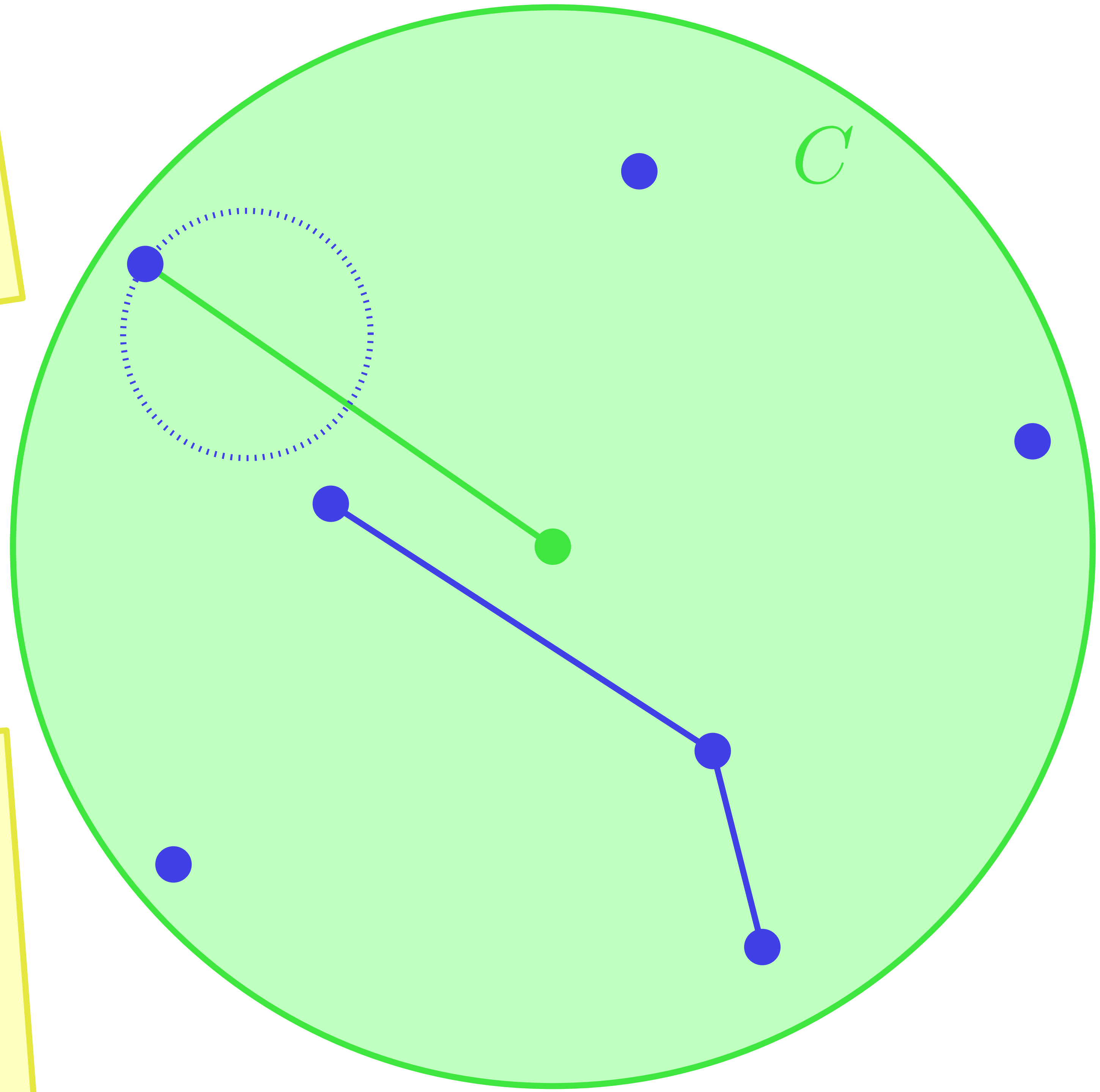
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.

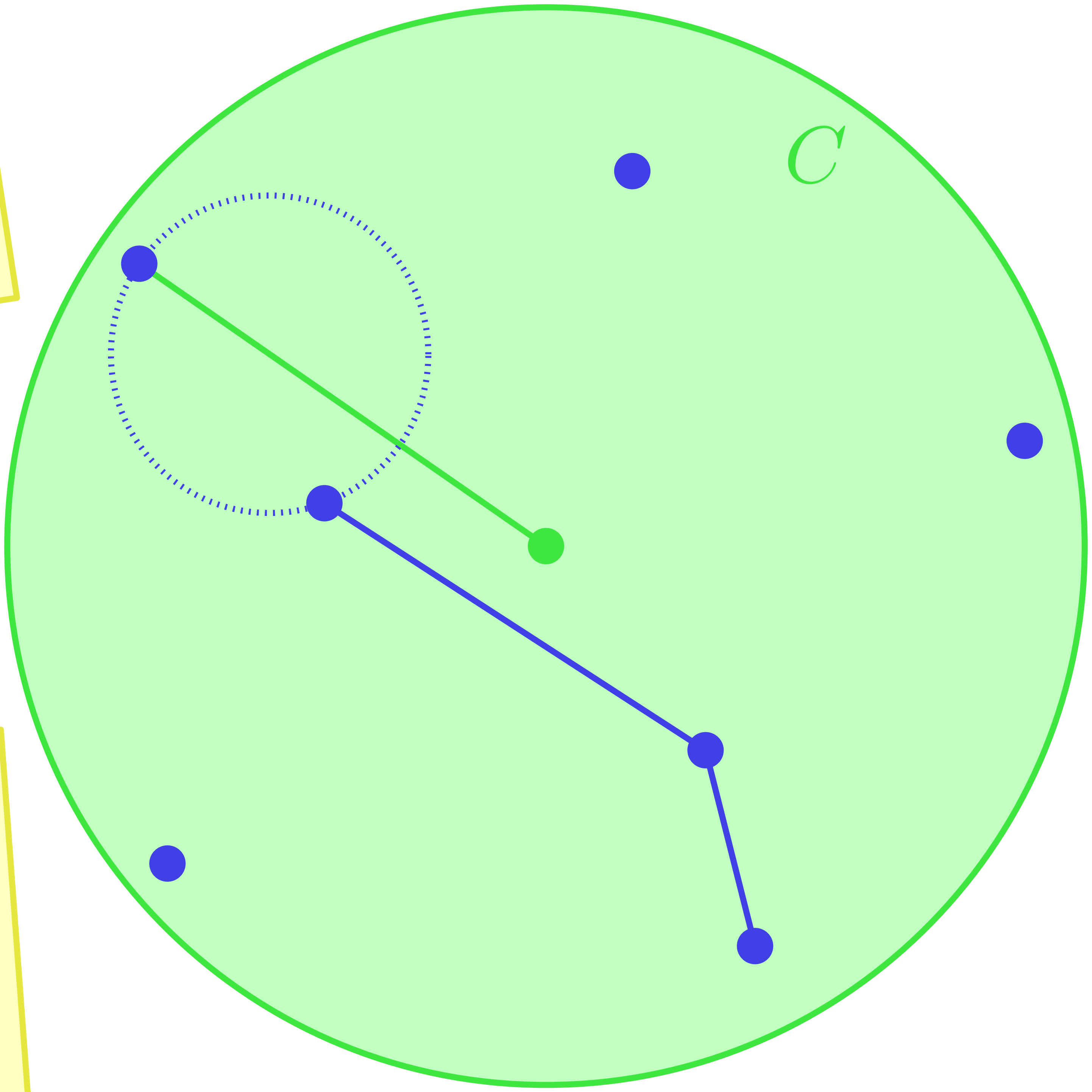
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw some circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

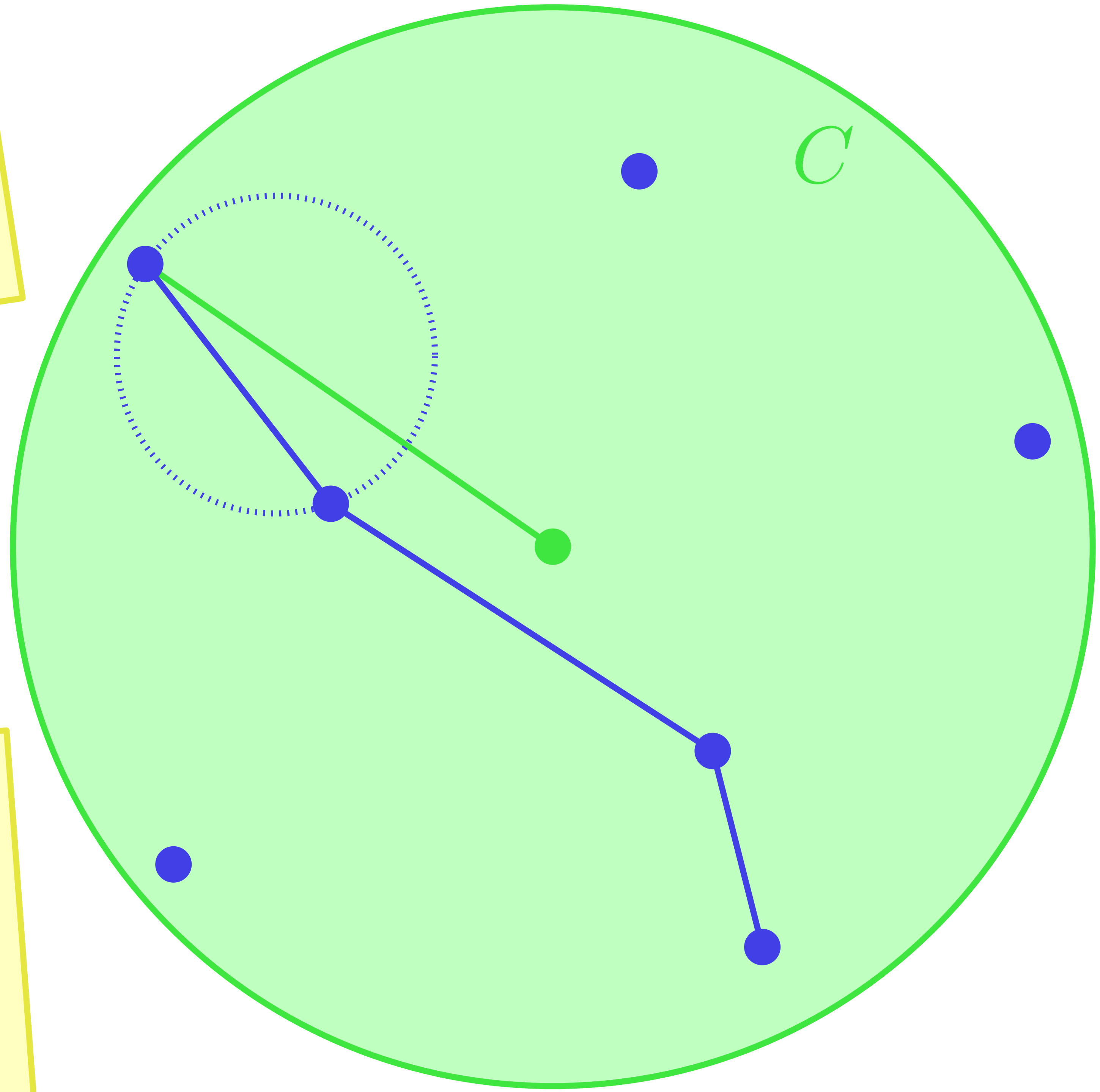
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

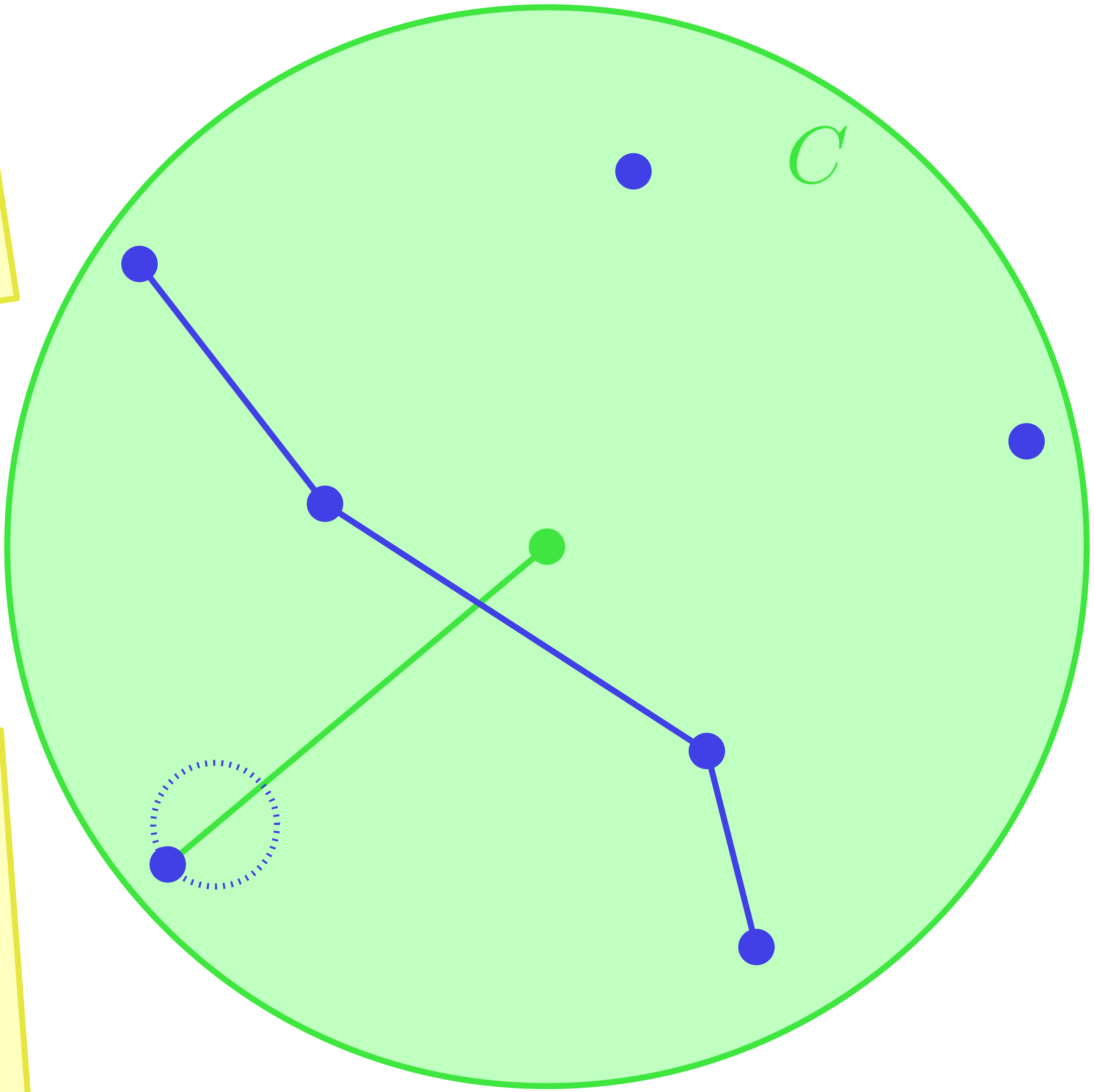
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw [...]me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.

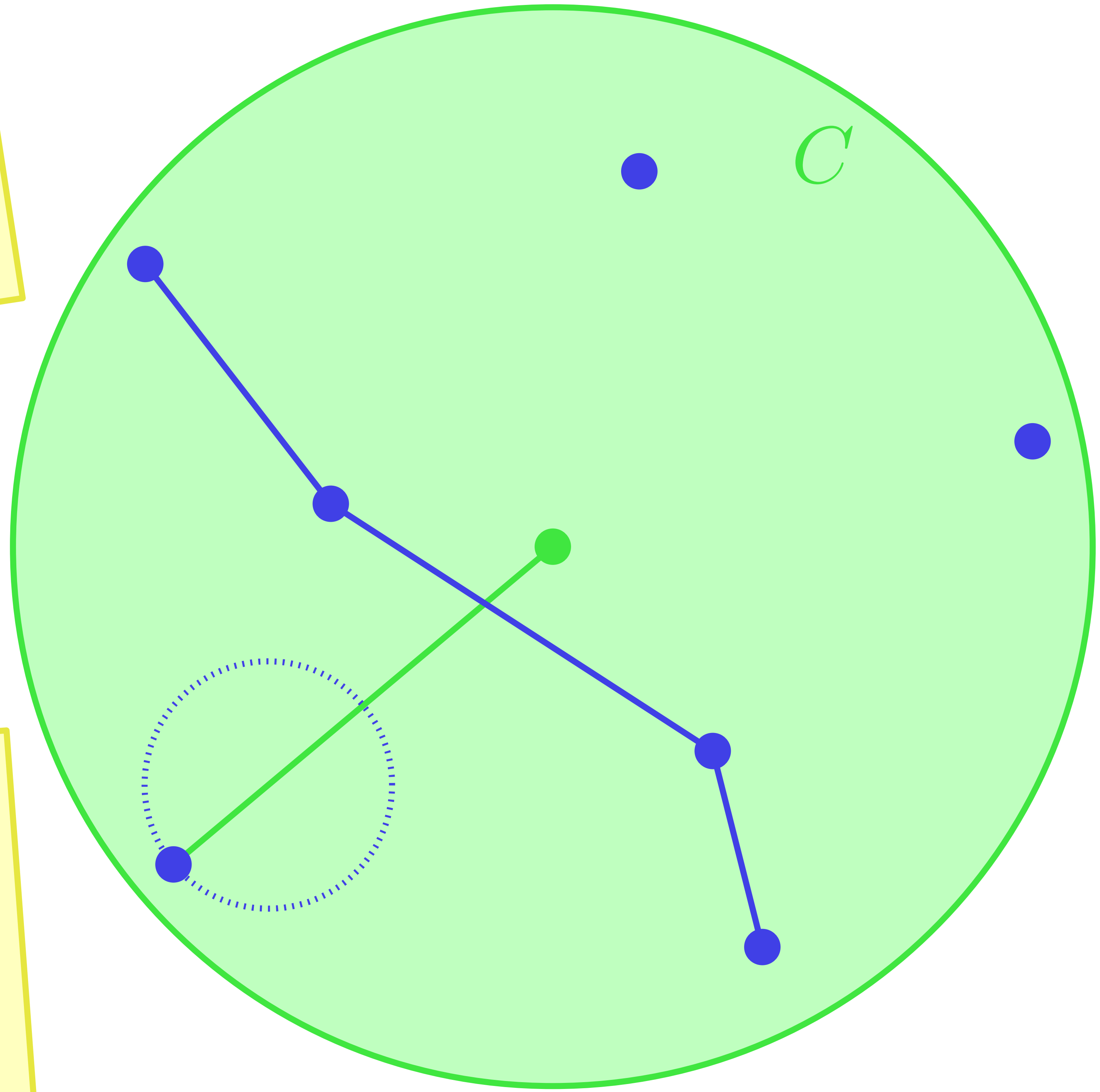
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

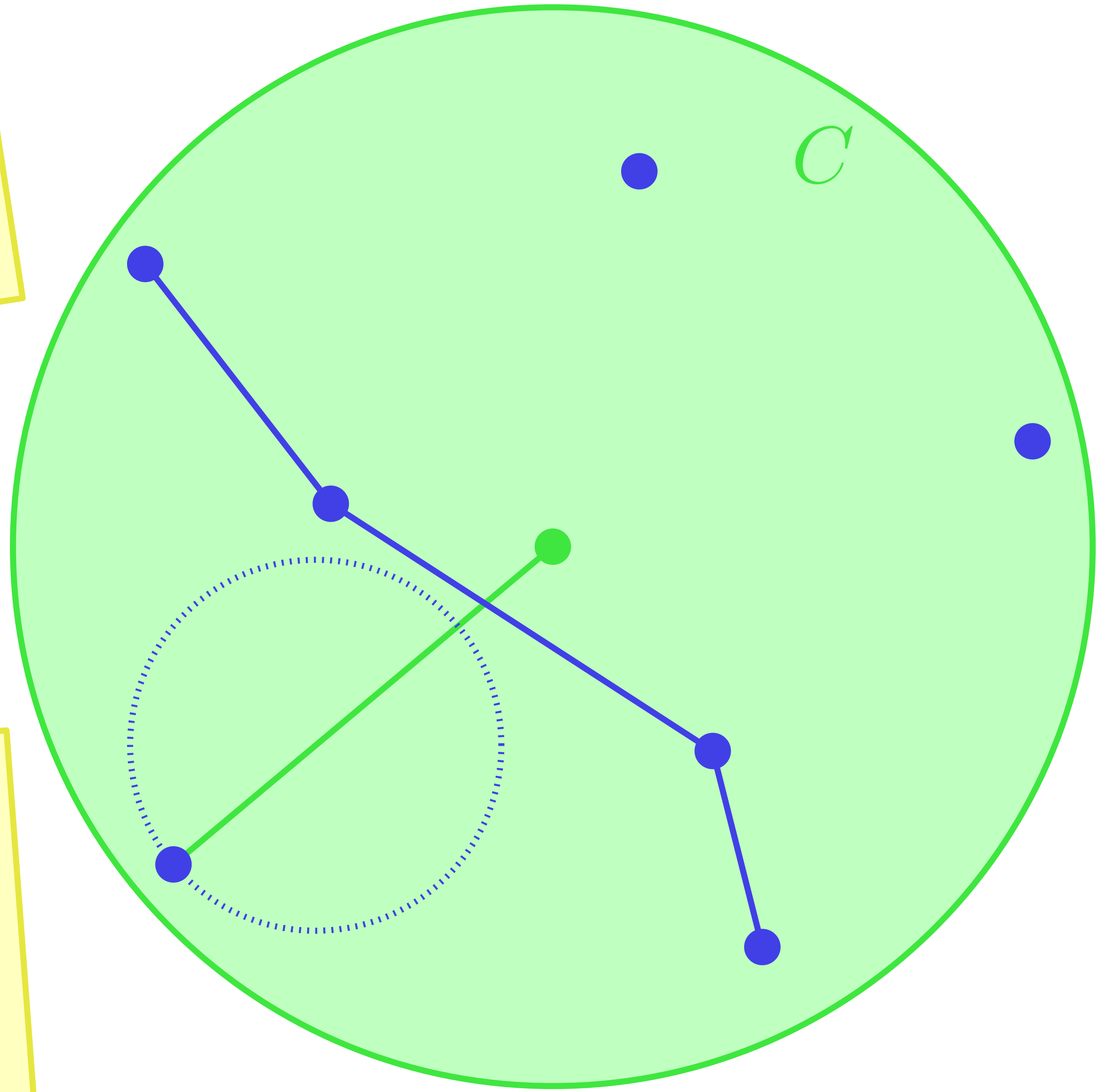
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

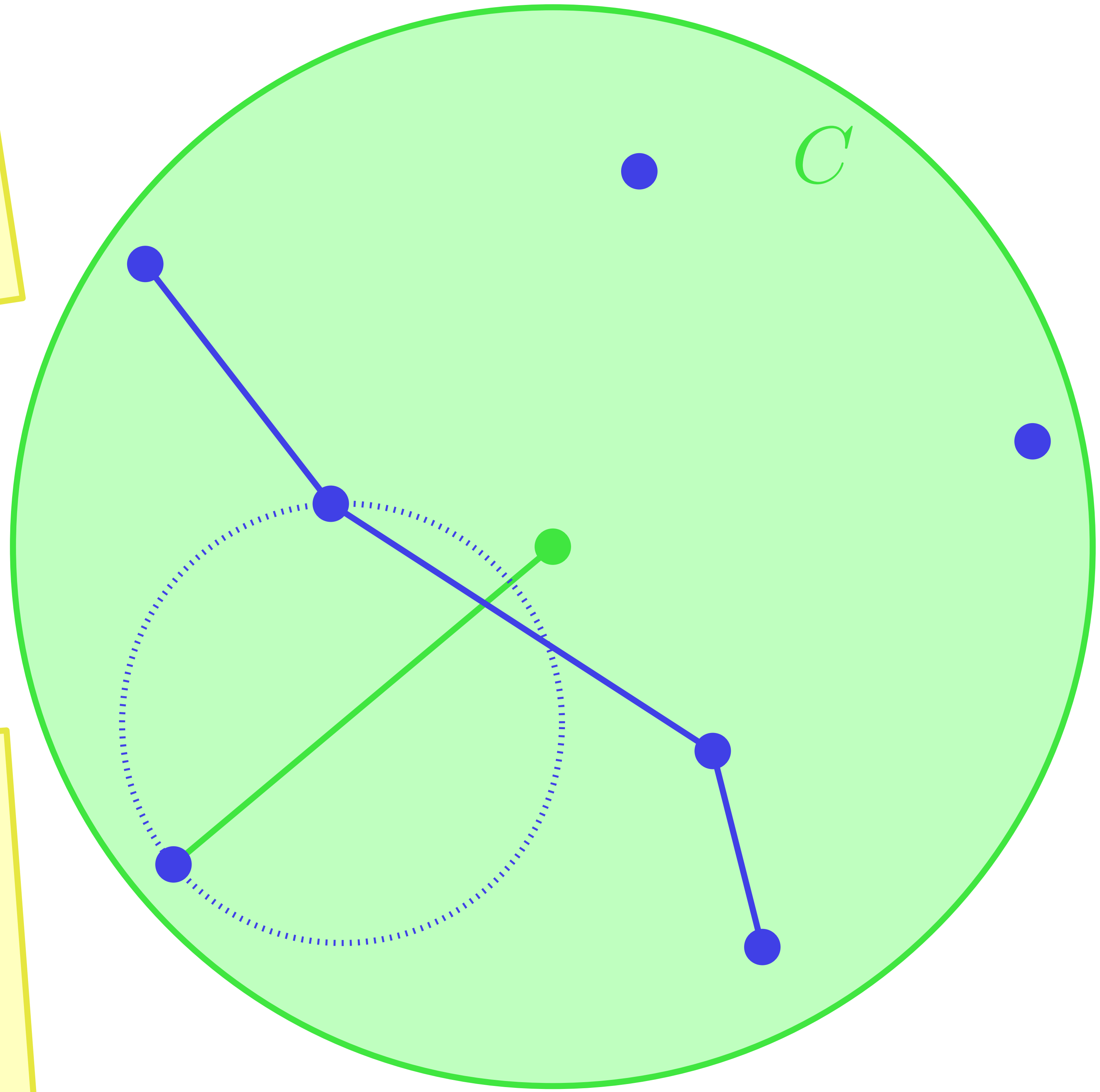
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.

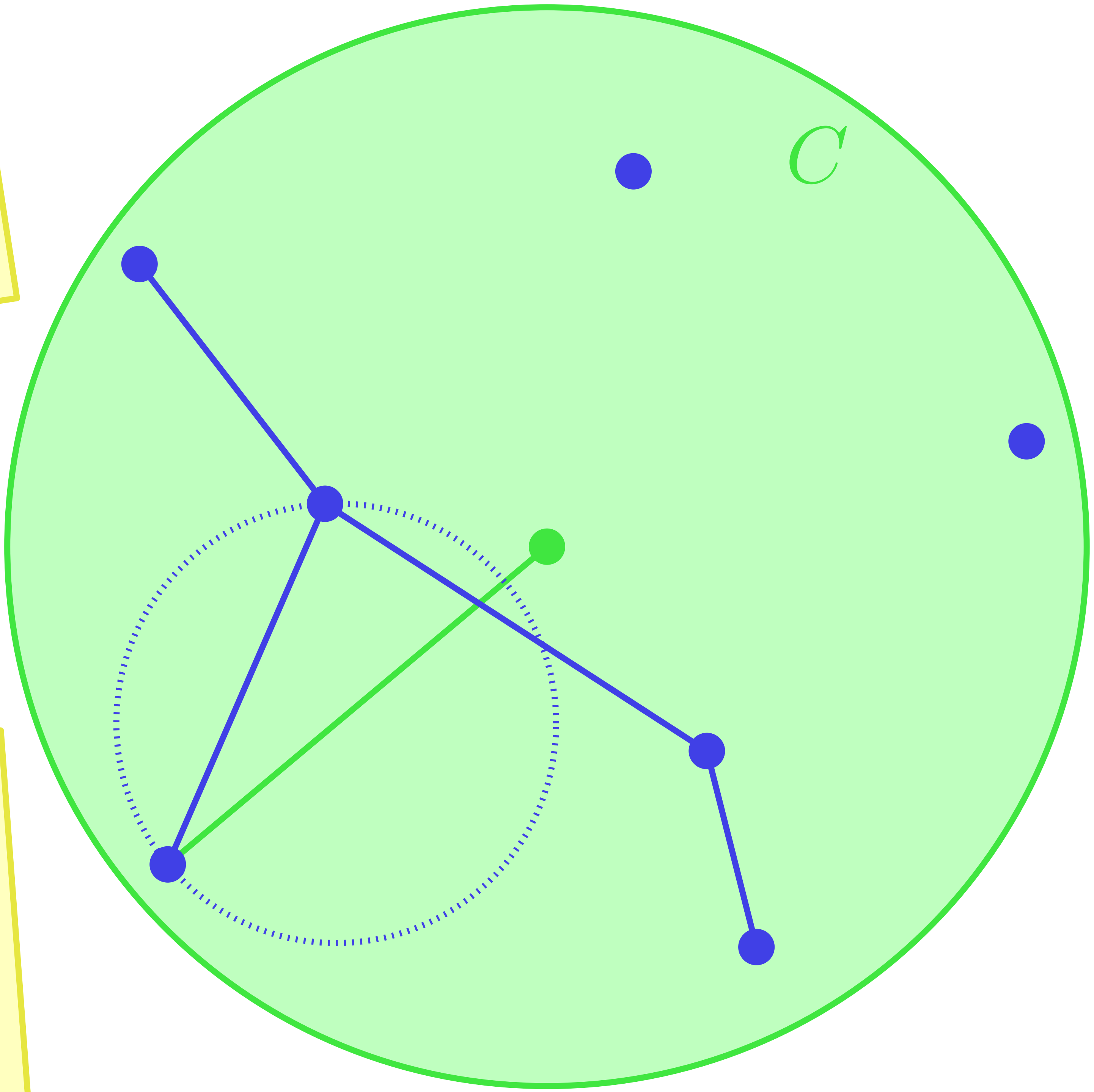
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.

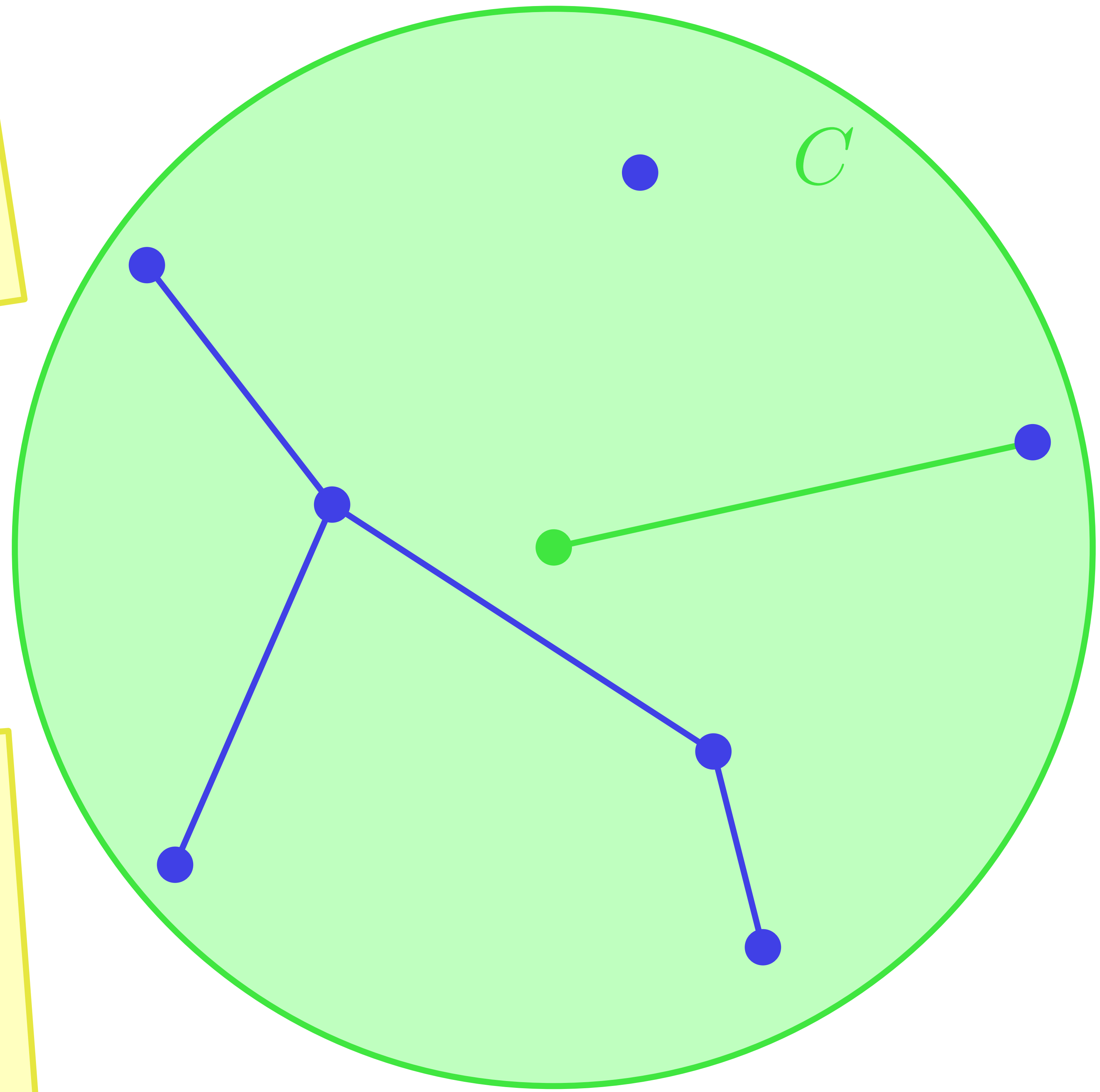
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

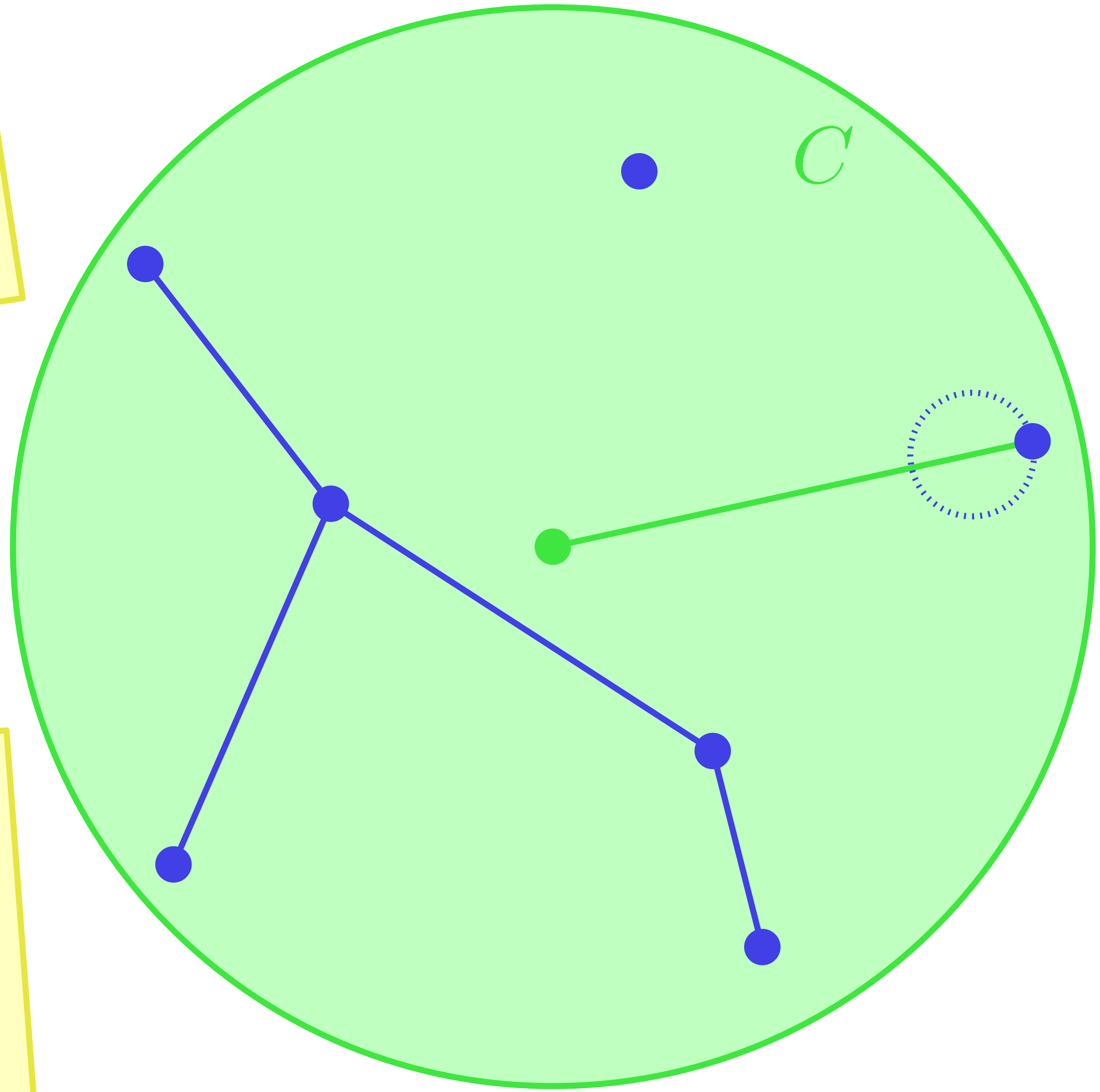
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

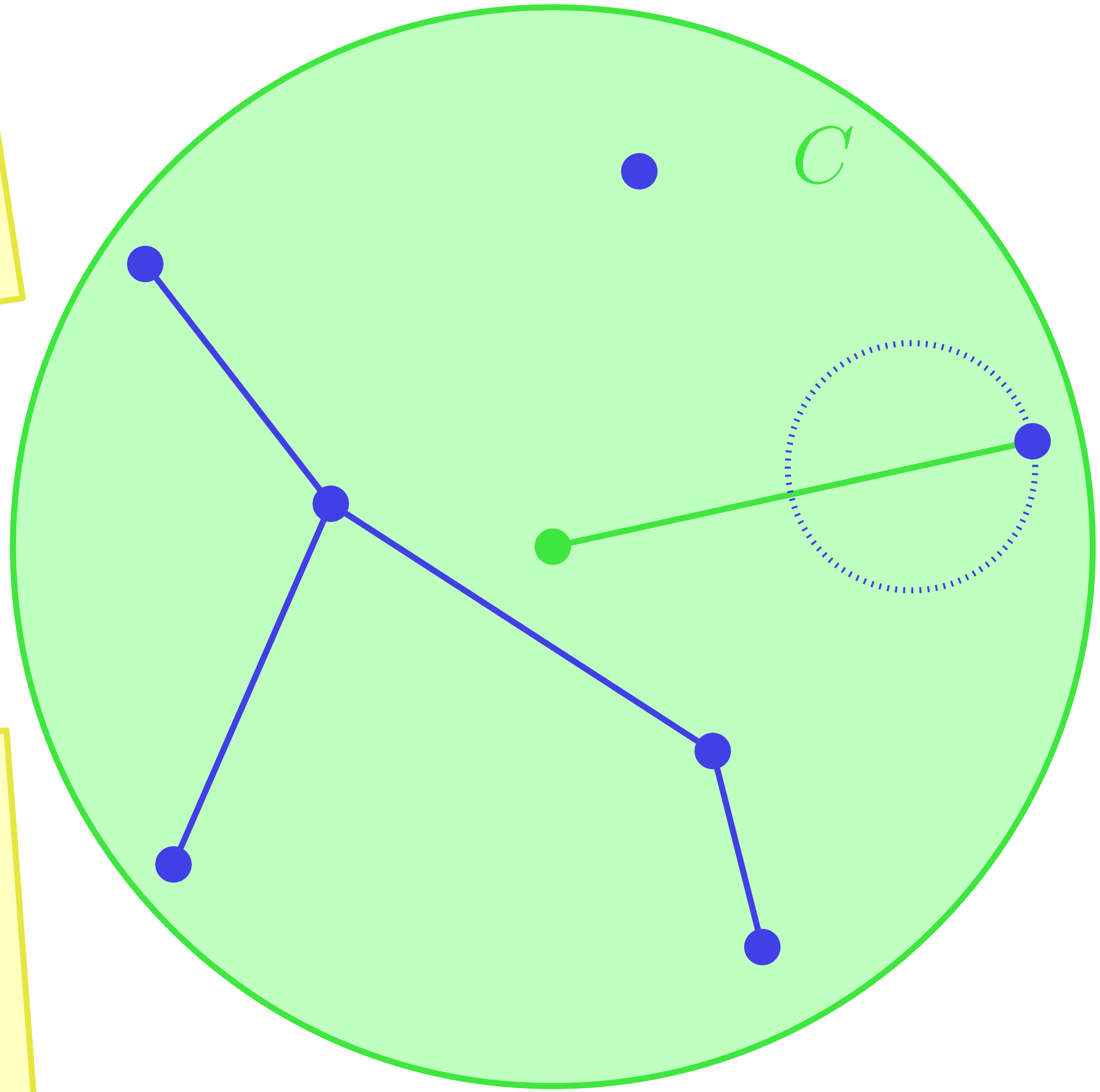
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

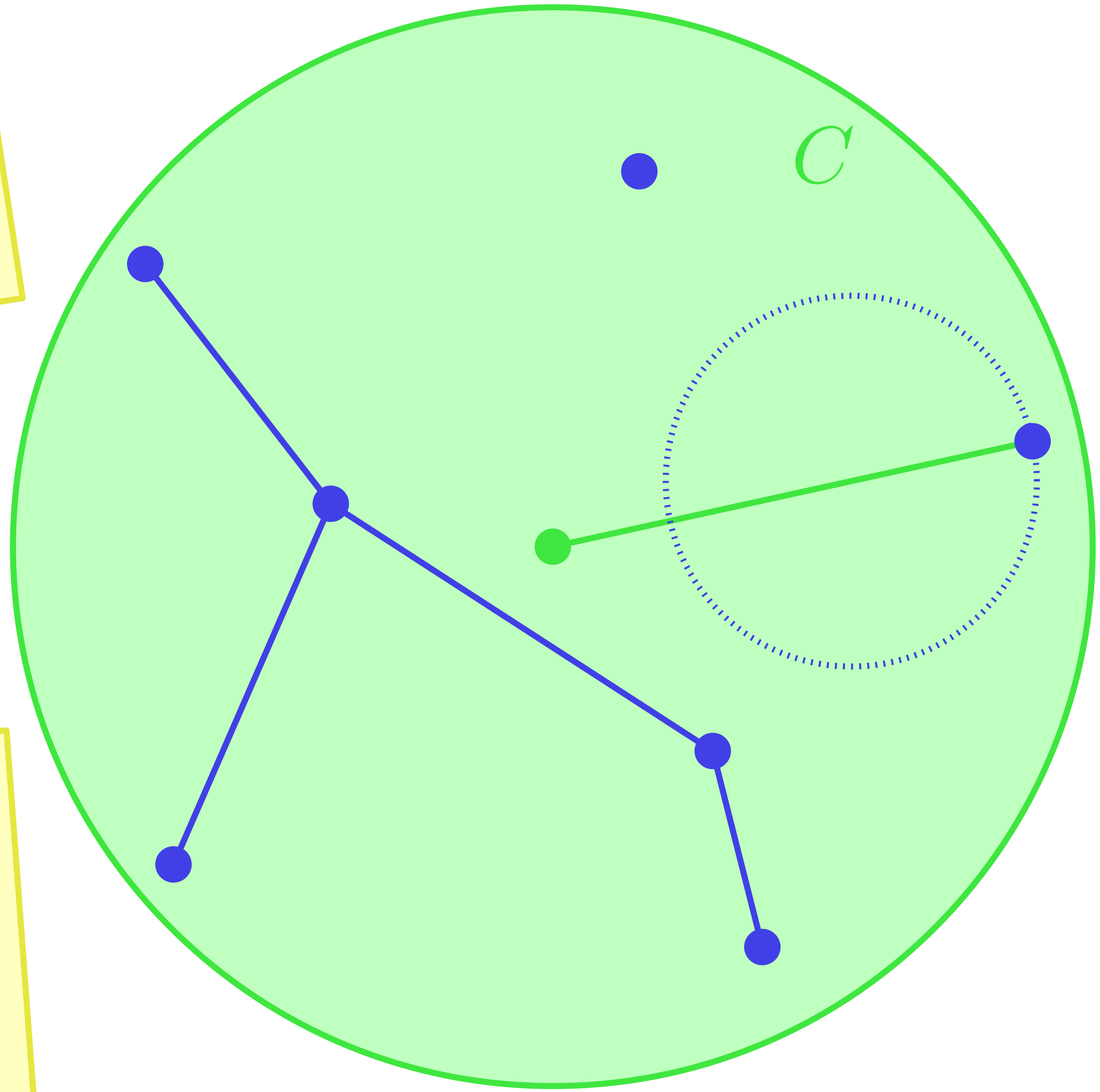
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.

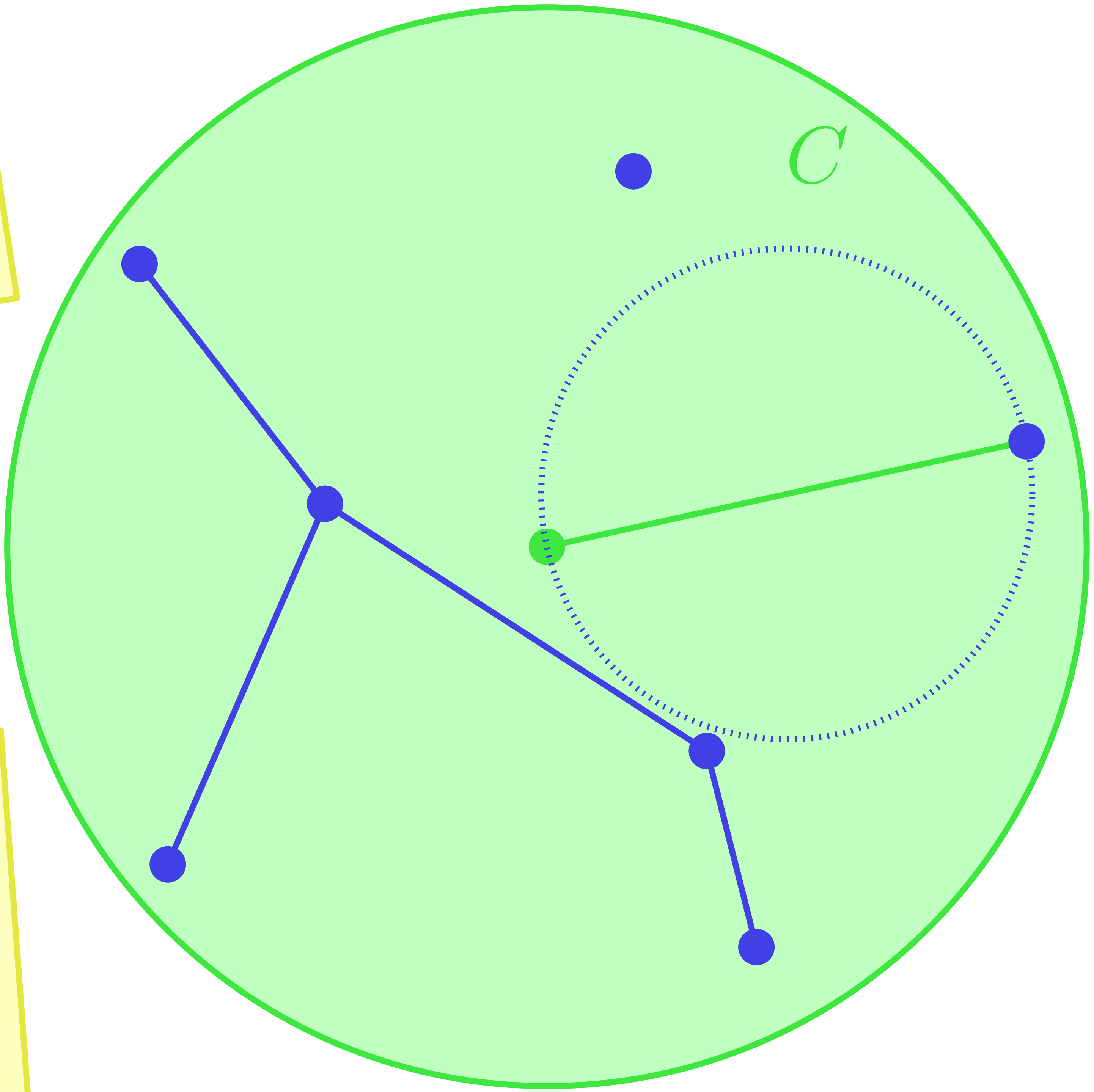
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

LEMMA
The resulting graph is always connected.

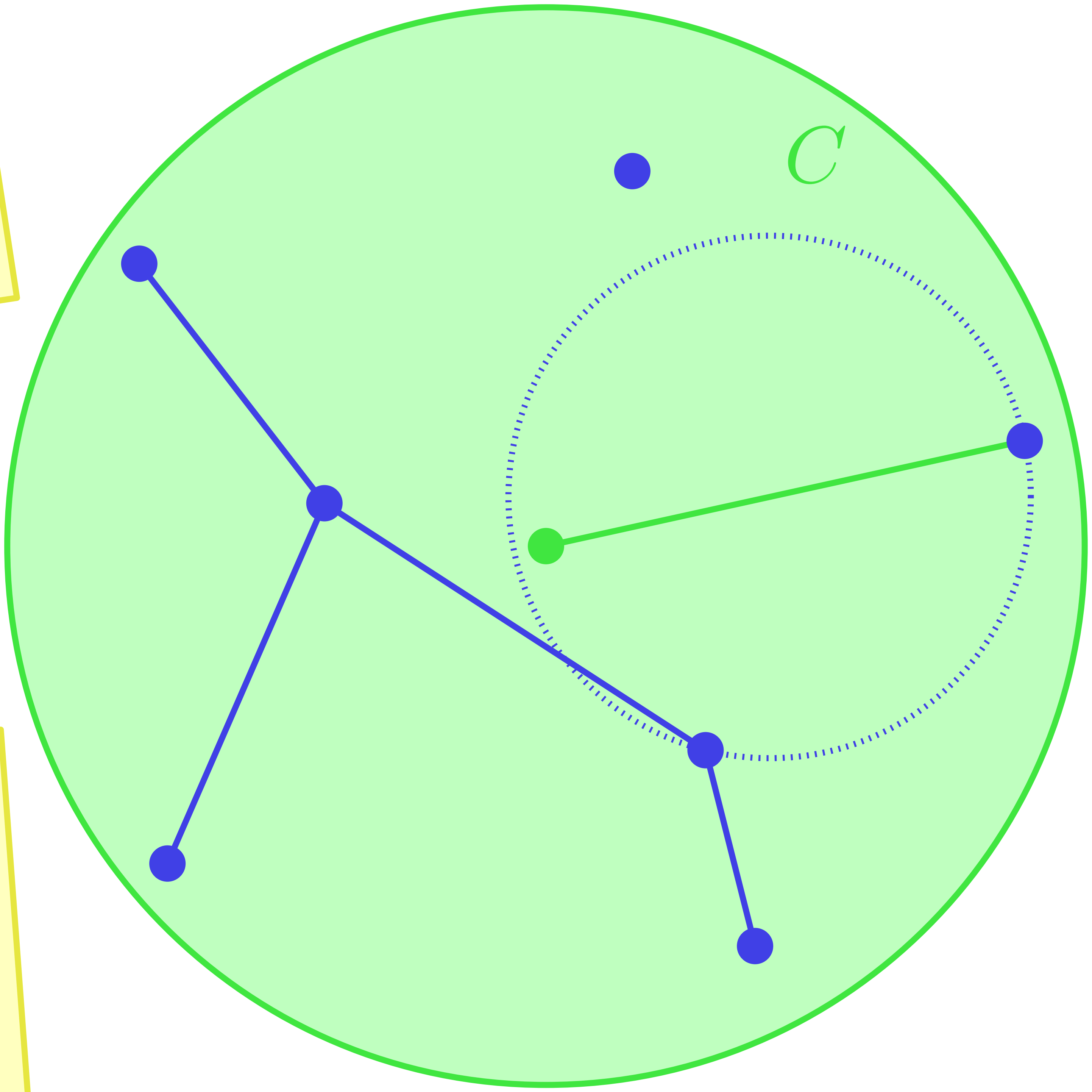


Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

LEMMA

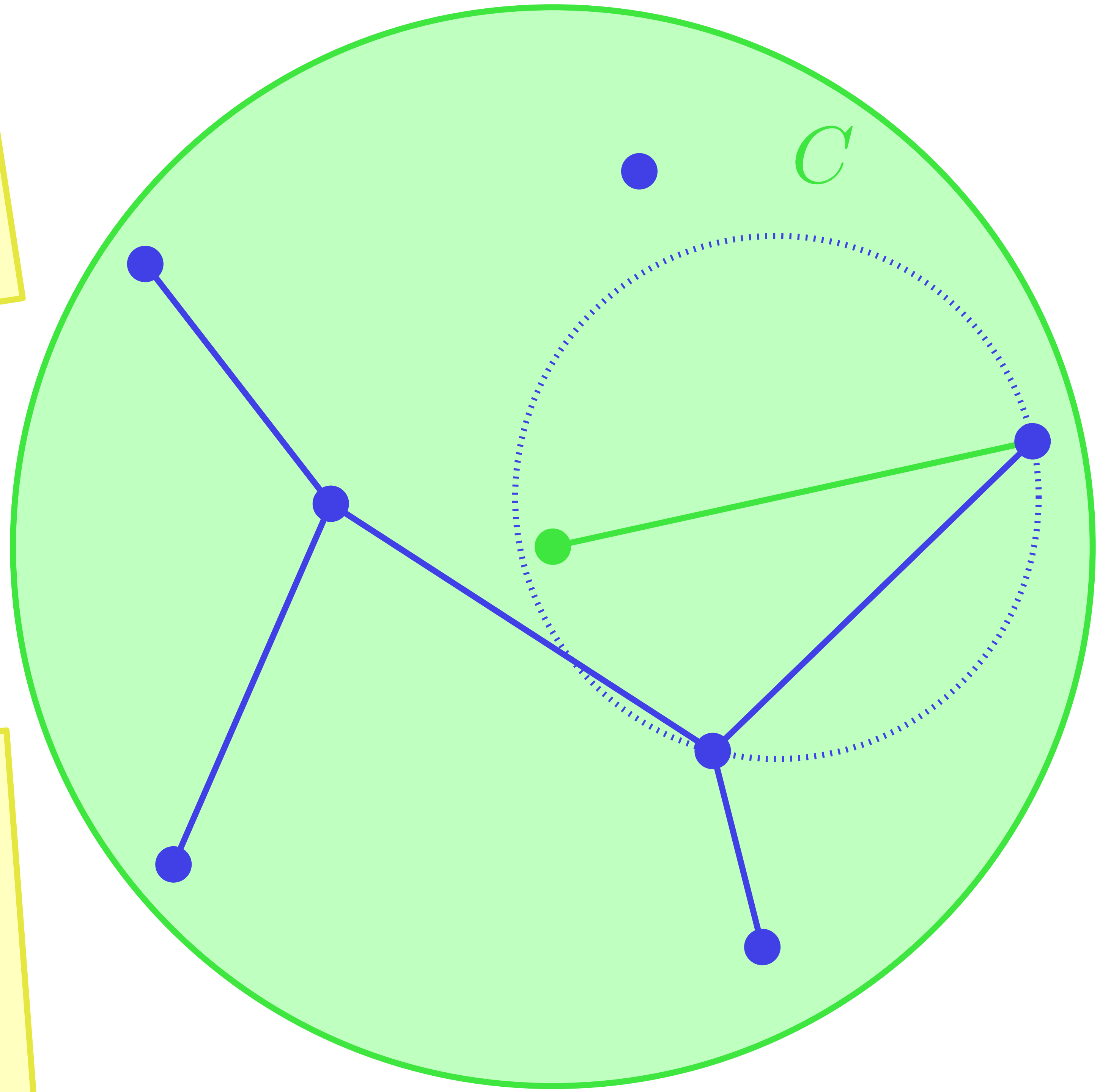
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

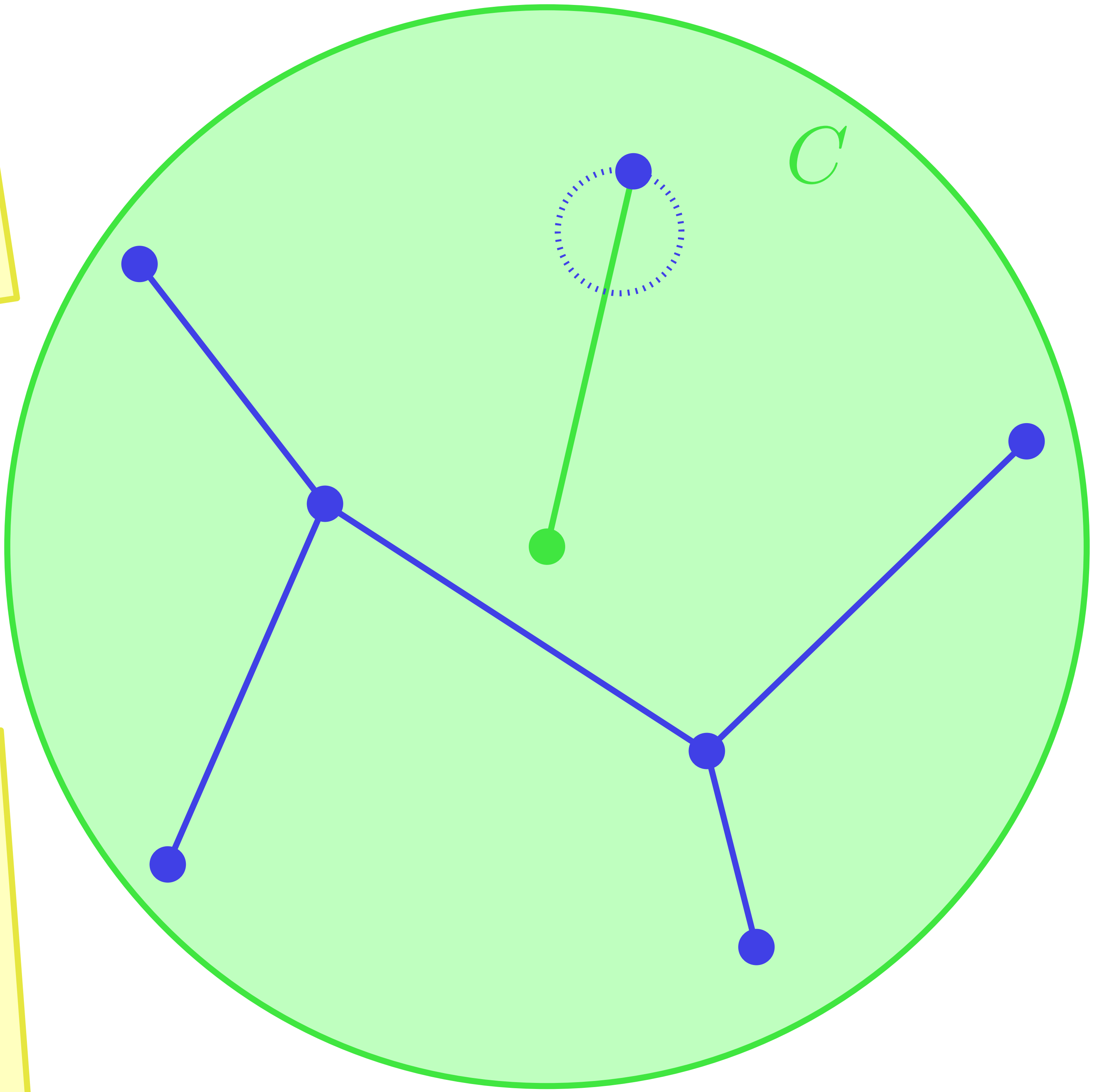
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.

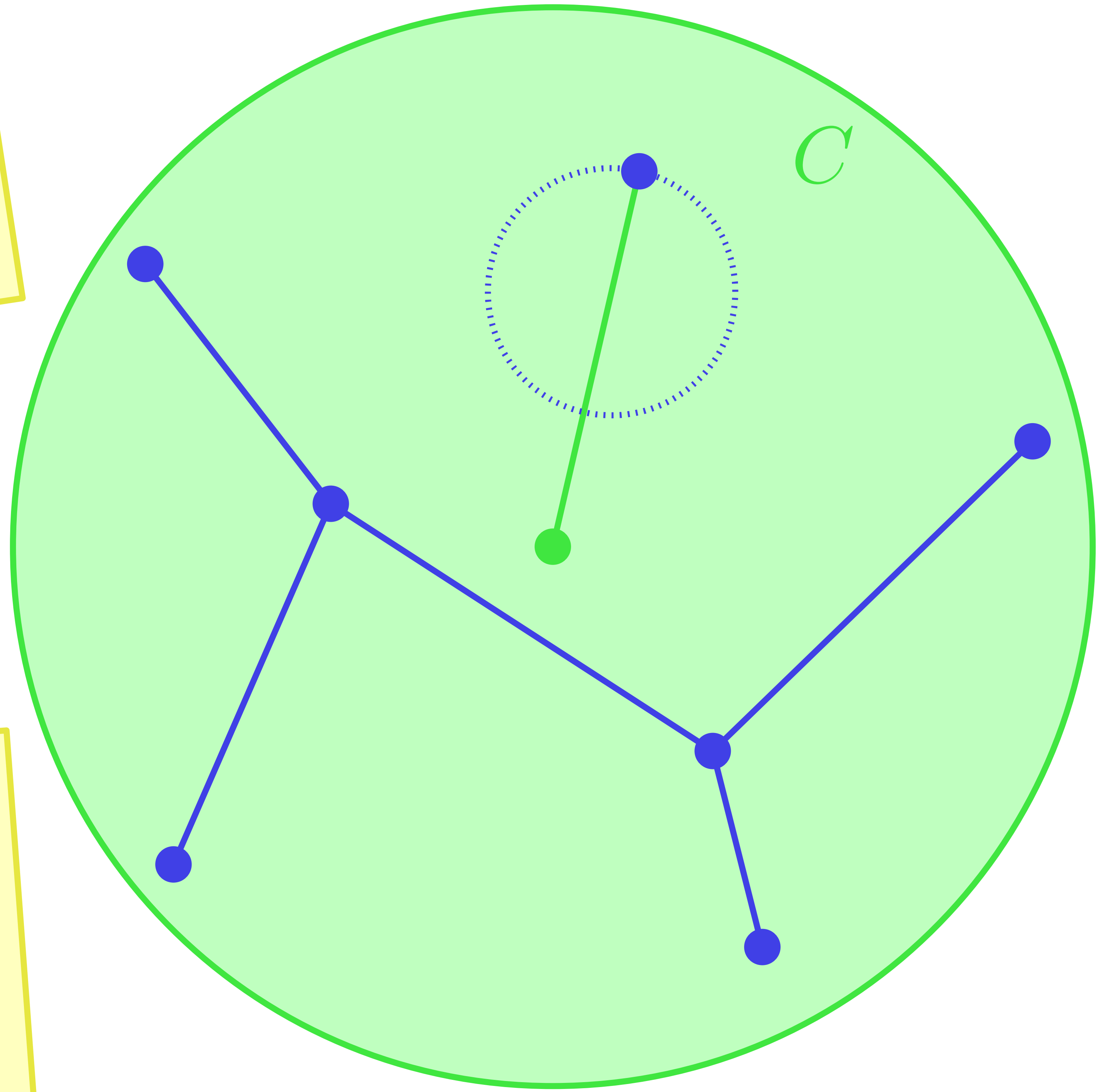
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.

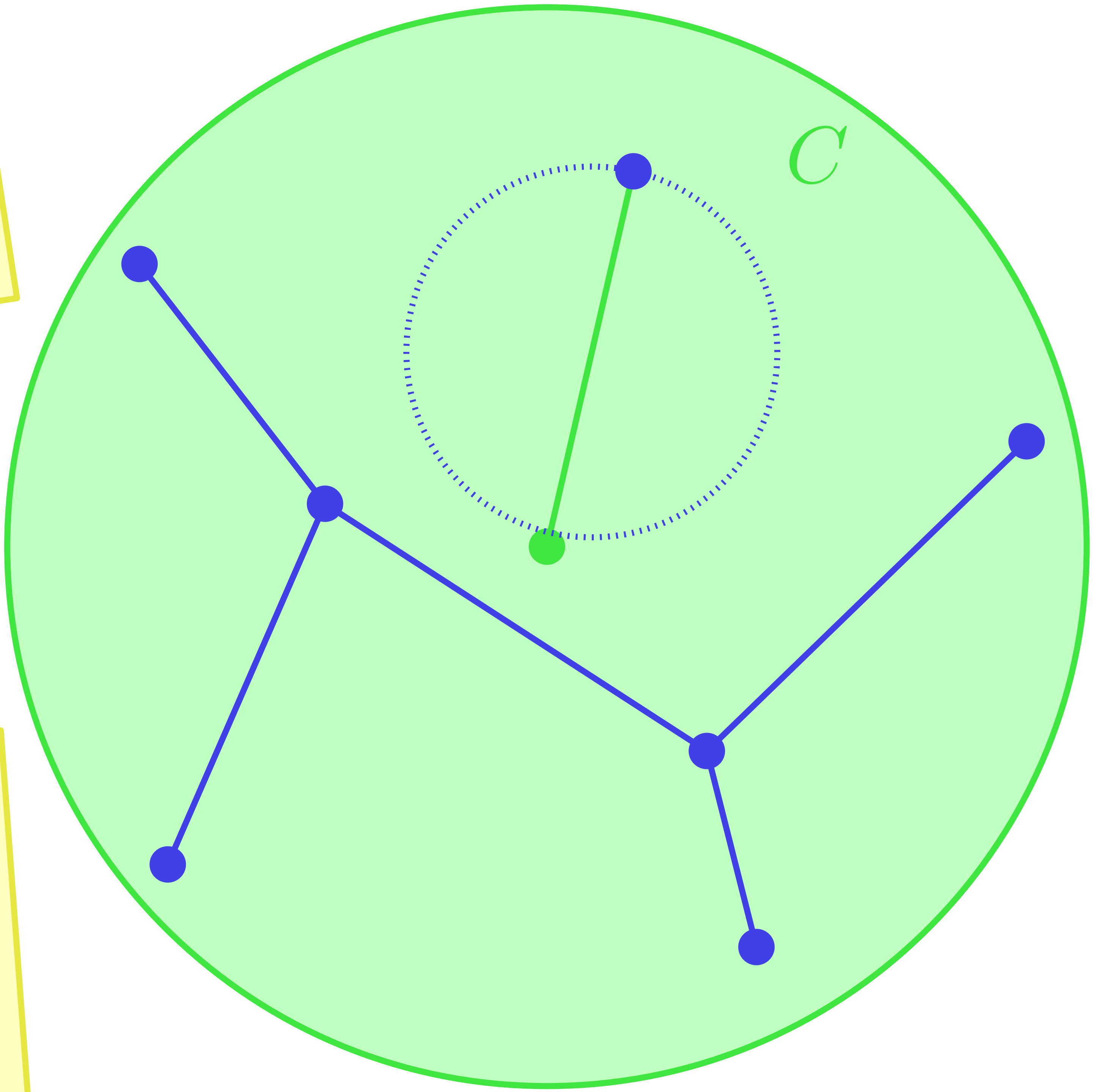
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.

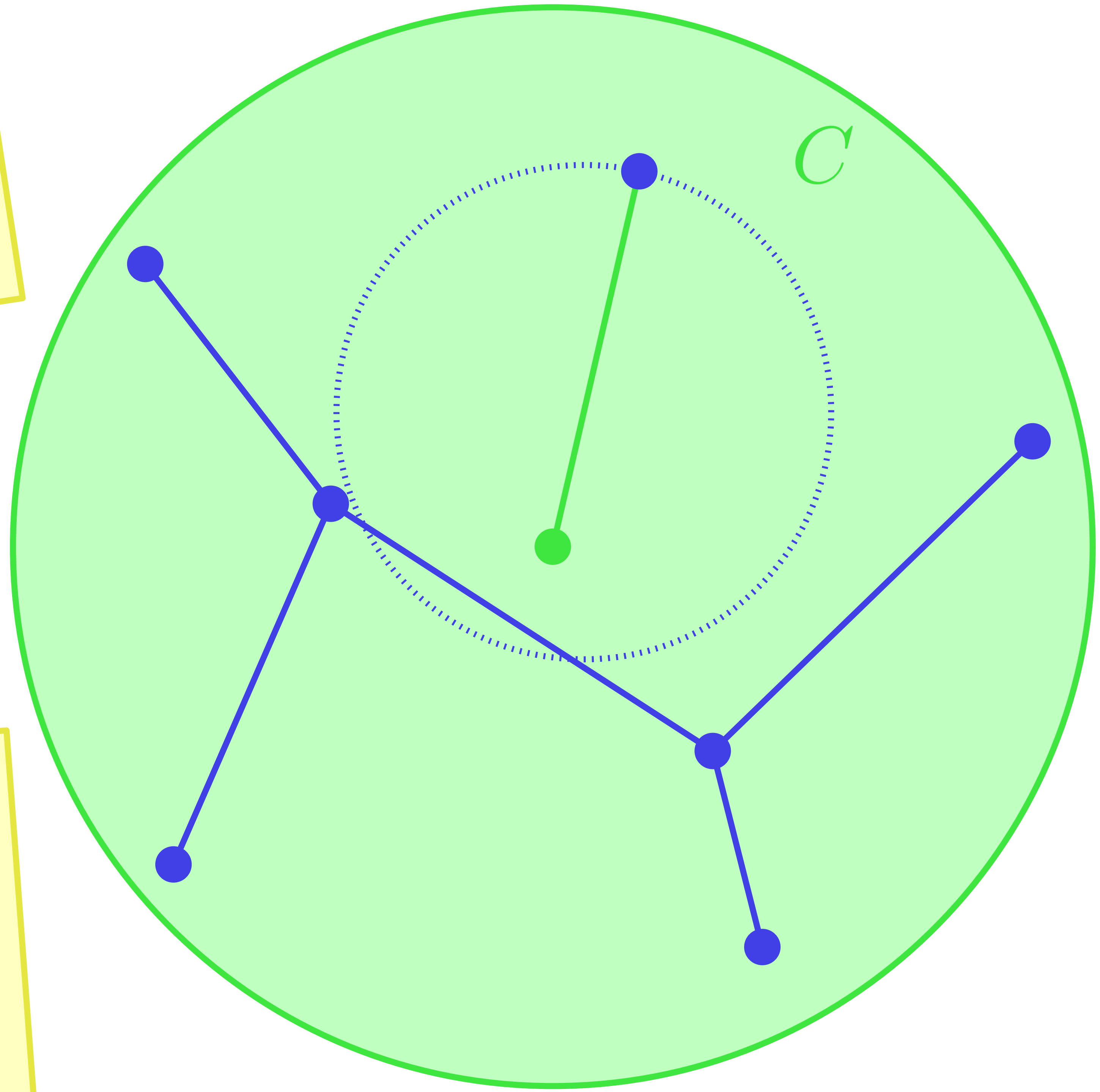
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw
me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.

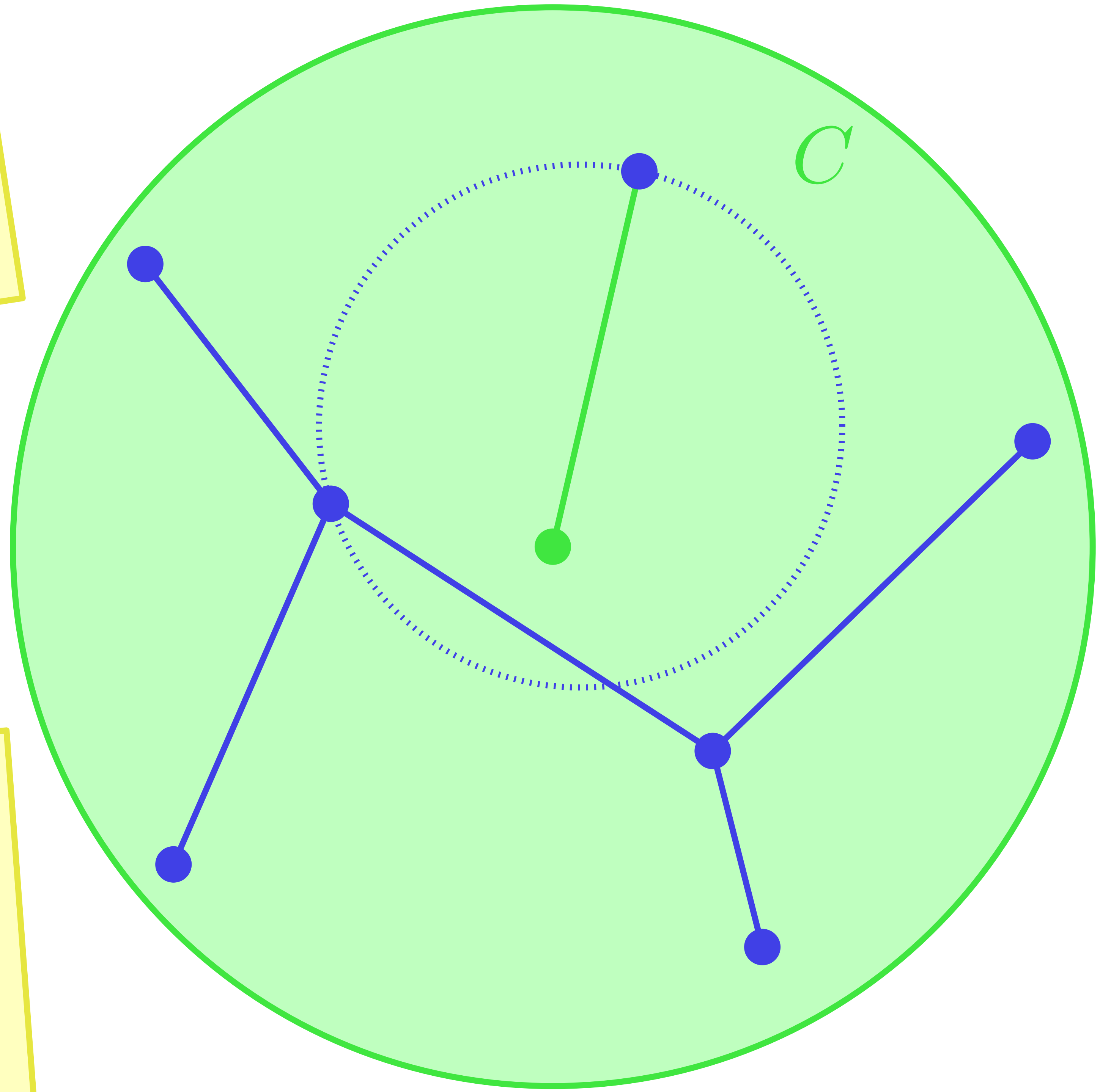
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw ...me circle C.

An edge e of T is certified by C if there is an empty circle for e completely inside C.

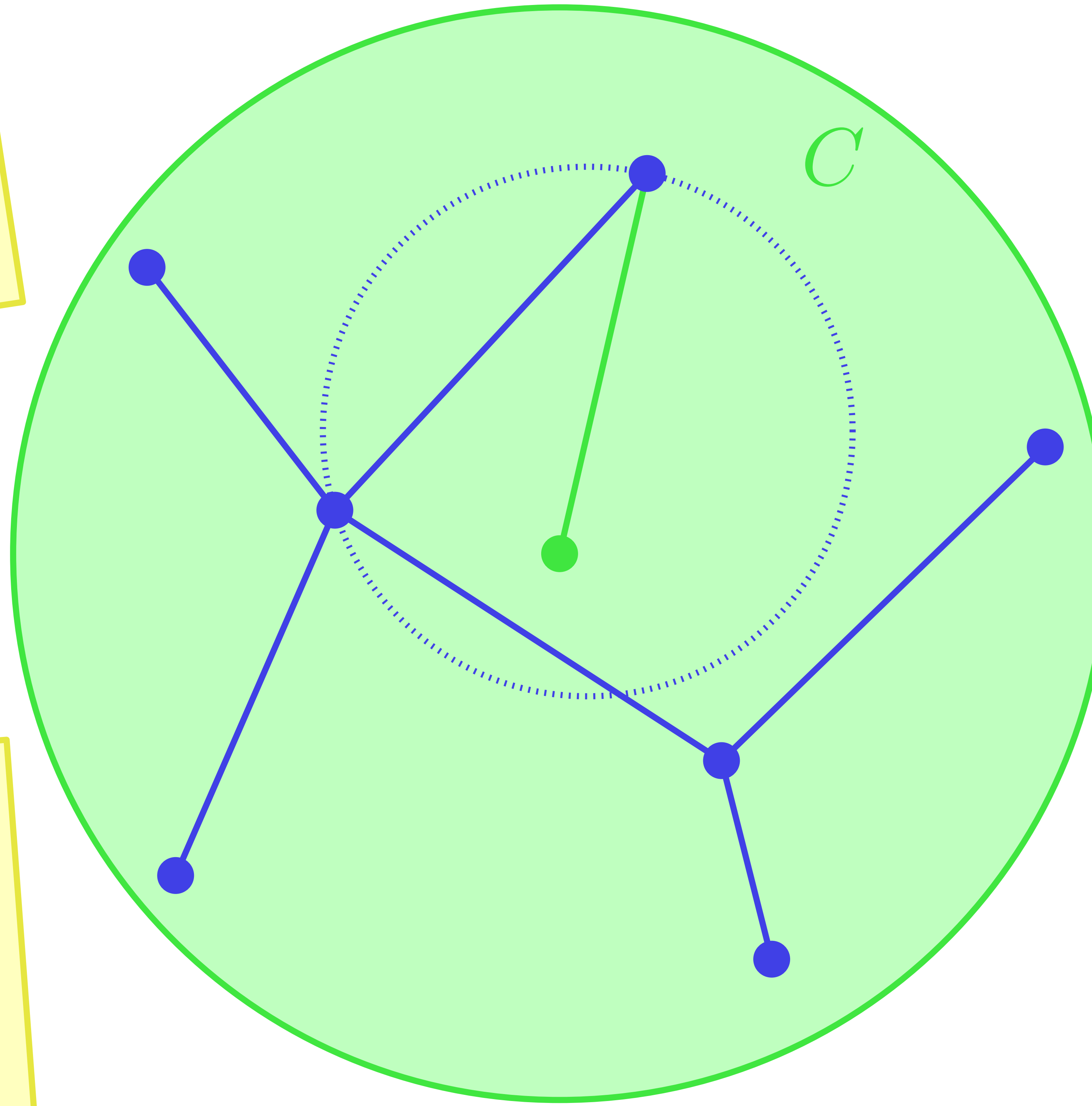
LEMMA
The resulting graph is always connected.



Consider a point set P, its Delaunay triangulation T, and draw …me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

LEMMA
The resulting graph is always connected.

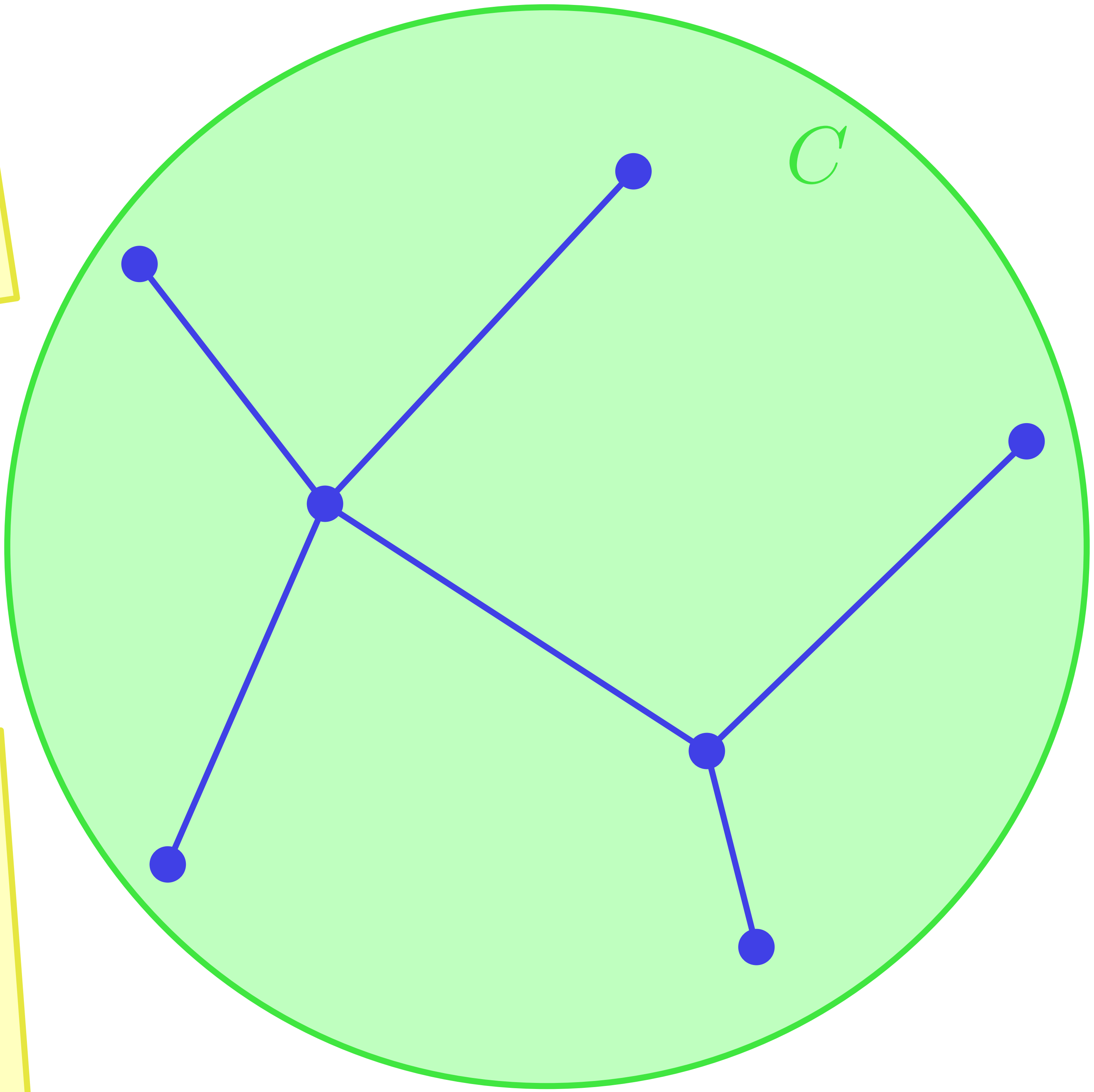


Consider a point set P, its Delaunay triangulation T, and draw me circle C.

An edge e of T is *certified* by C if there is an empty circle for e completely inside C.

LEMMA

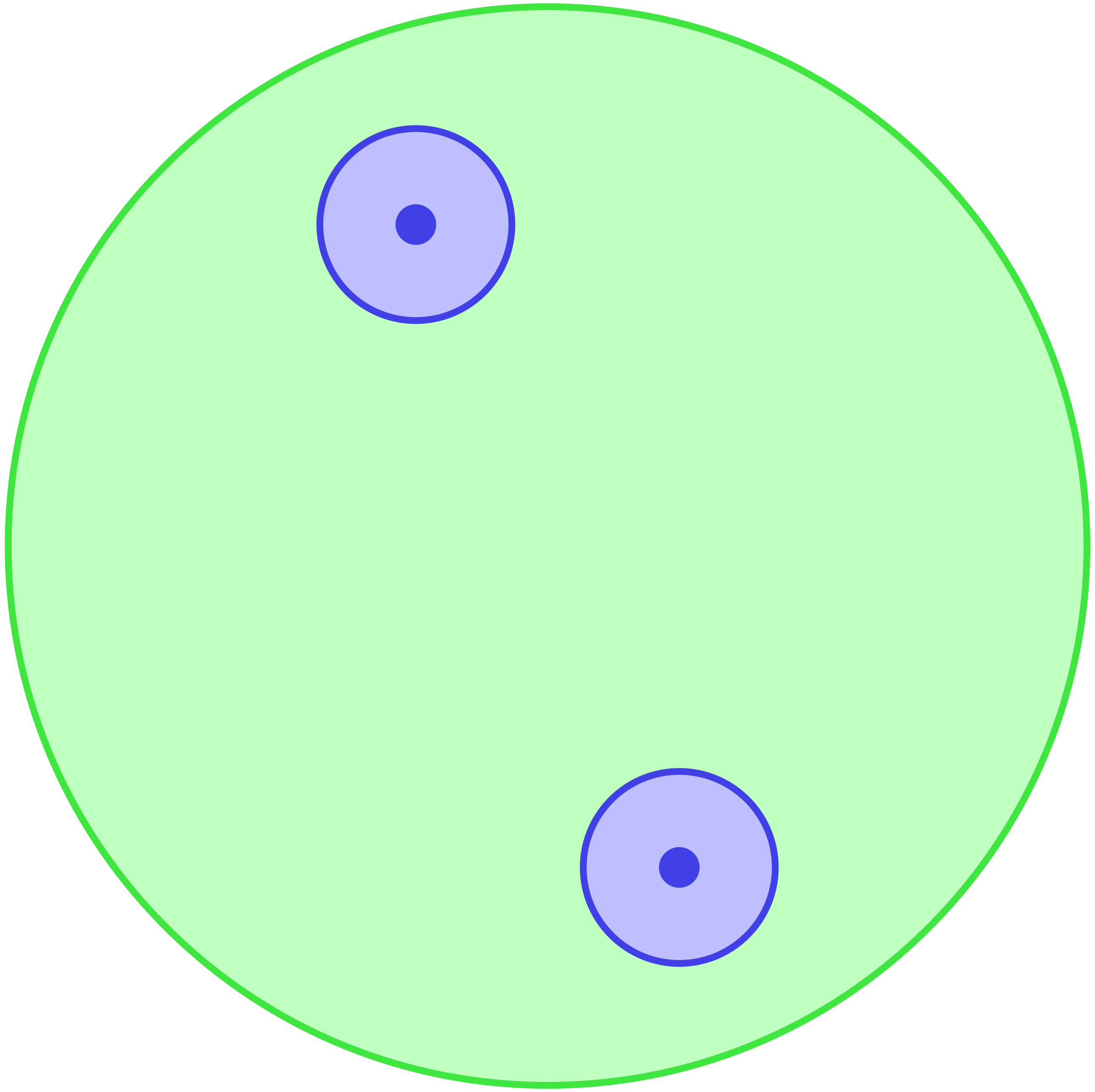
The resulting graph is always connected.



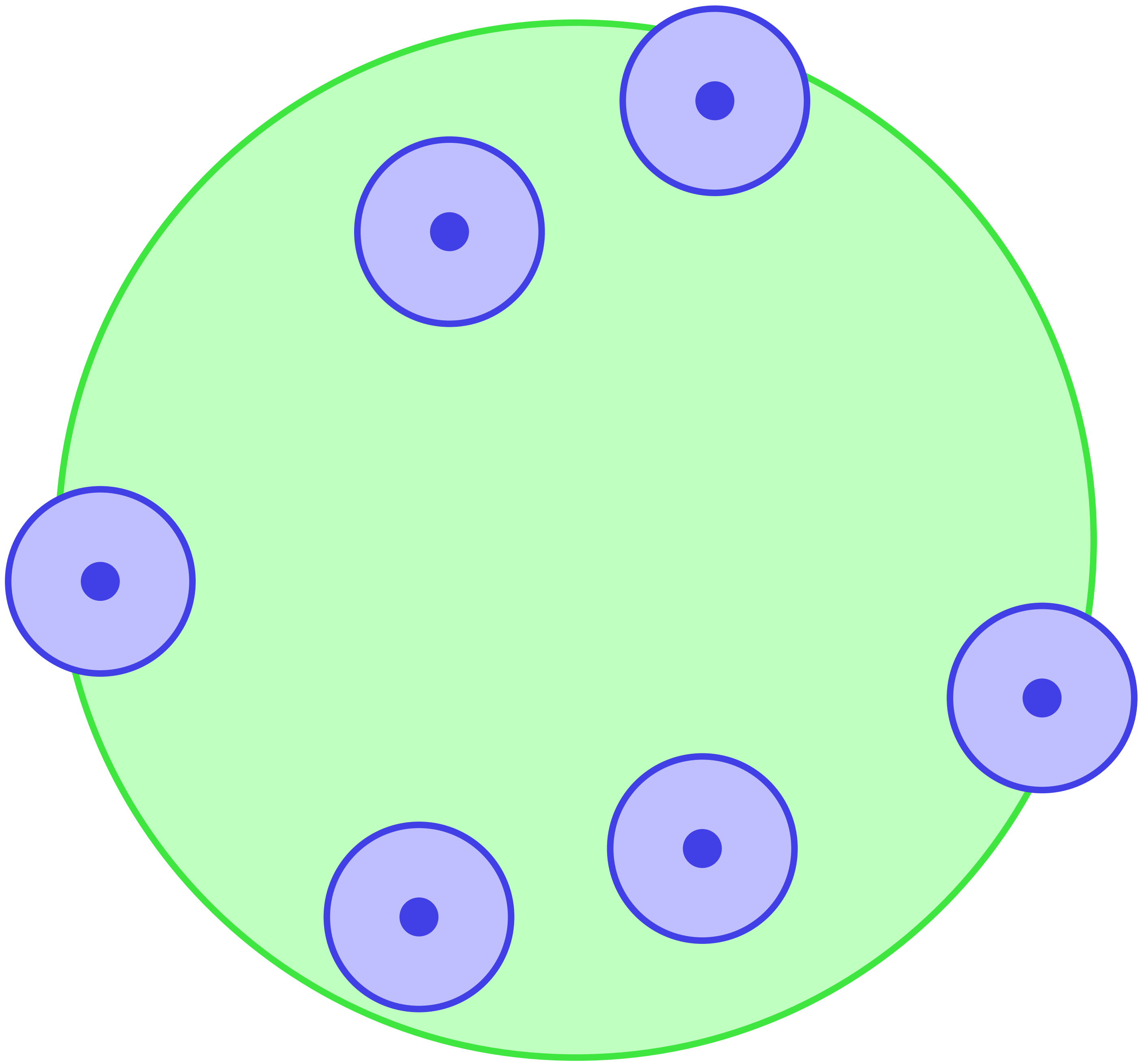
So, how do we reconstruct a spanning tree edge?

We reconstruct connectivity of the MST edges of P by length (increasing).

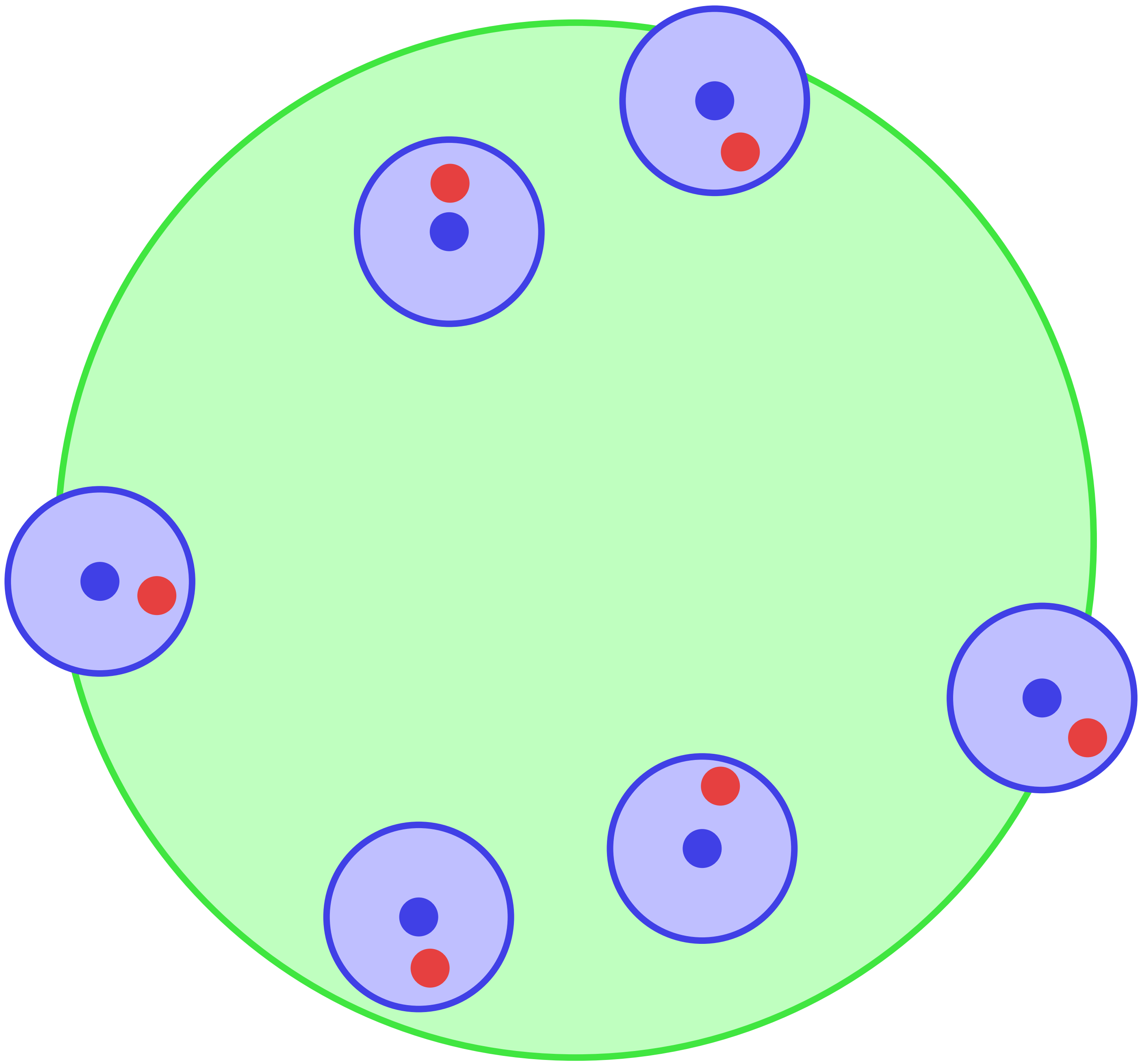
We reconstruct connectivity of the MST edges of P by length (increasing).



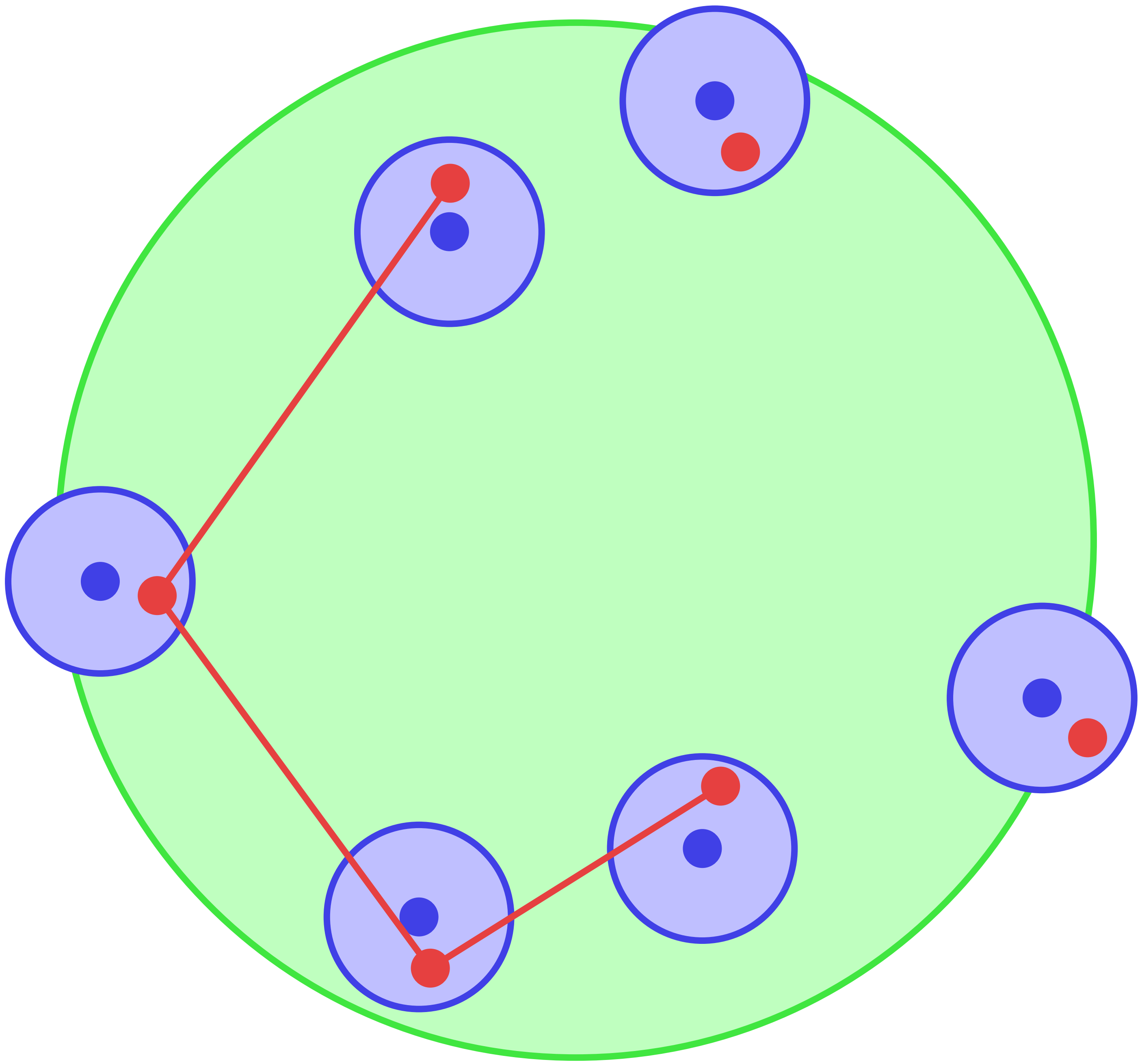
We reconstruct connectivity of the MST edges of P by length (increasing).



We reconstruct connectivity of the MST edges of P by length (increasing).

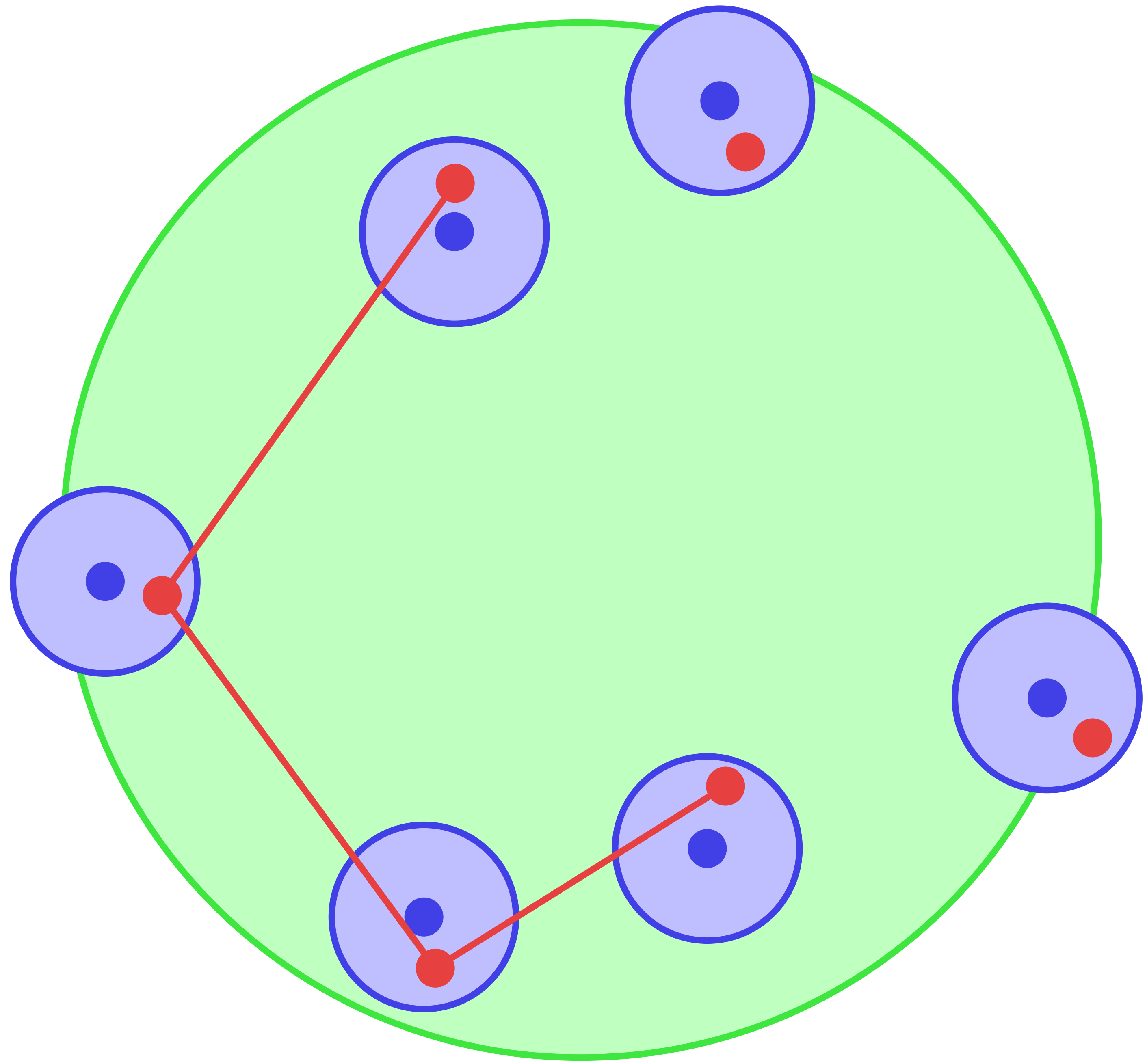


We reconstruct connectivity of the MST edges of P by length (increasing).



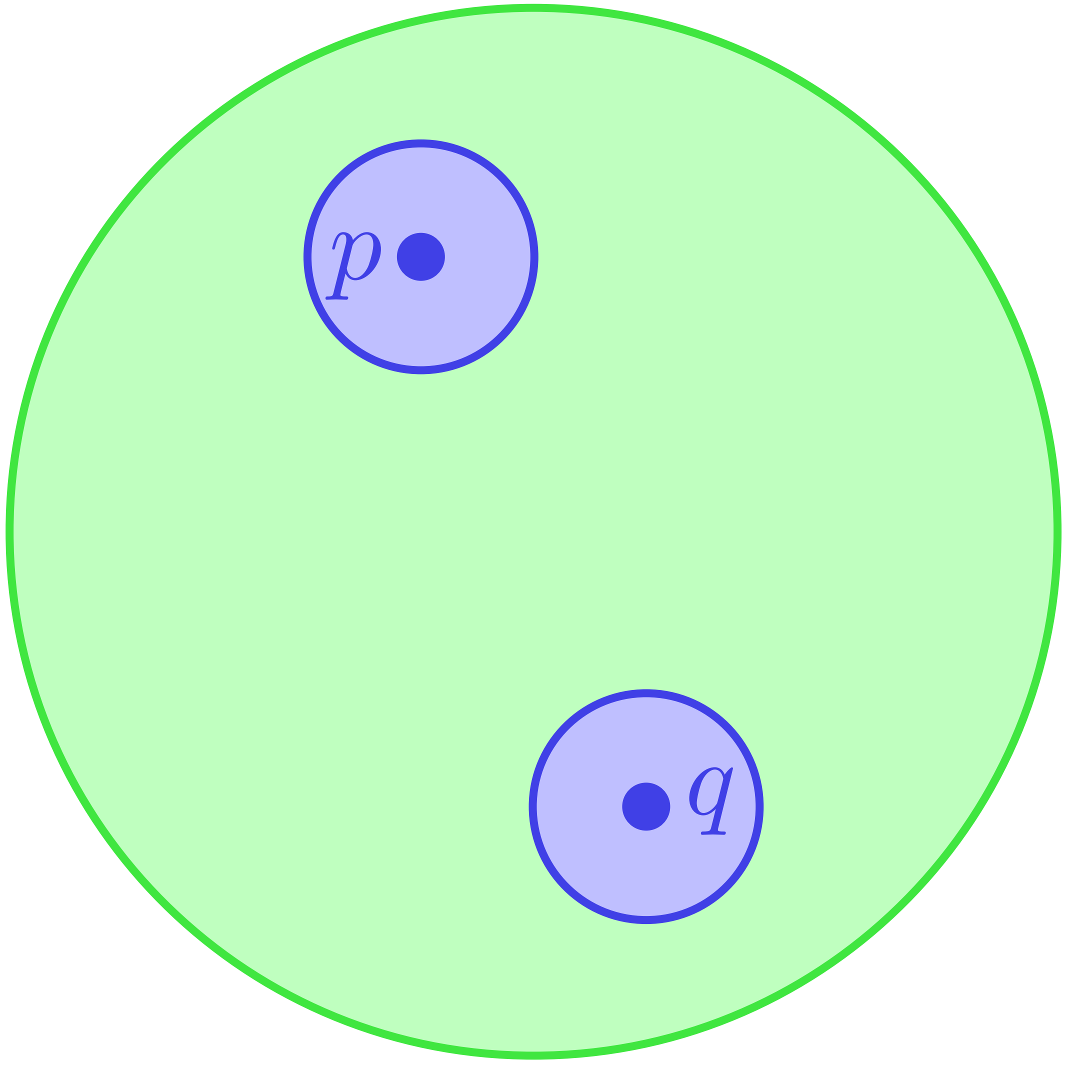
We reconstruct connectivity of the MST edges of P by length (incr

If an edge pq is *short*, the number of other points in C^+_{pq} is constant.



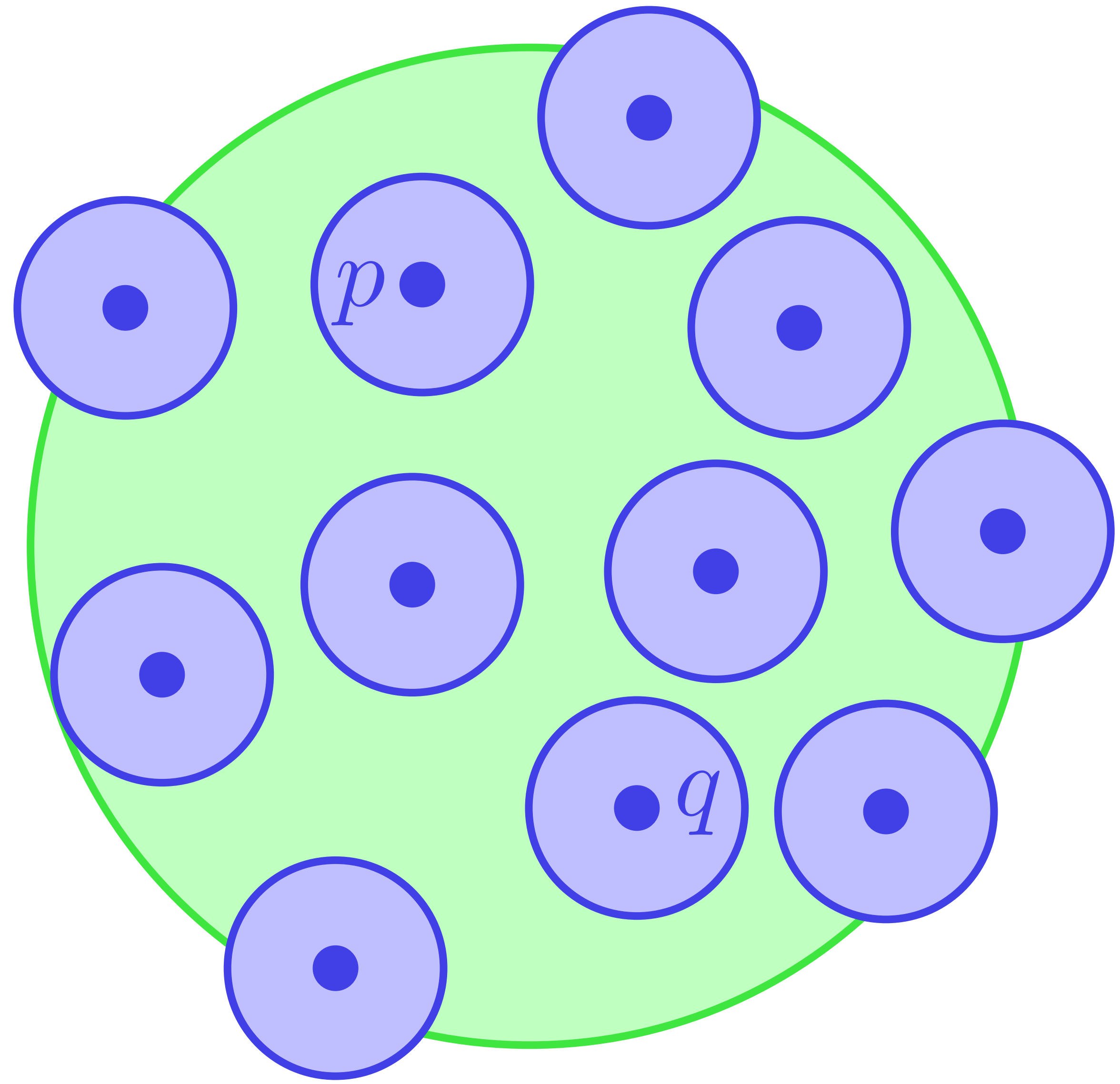
We reconstruct connectivity of the MST edges of P by length (incr

If an edge pq is *short*, the number of other points in C^{+}_{pq} is constant.



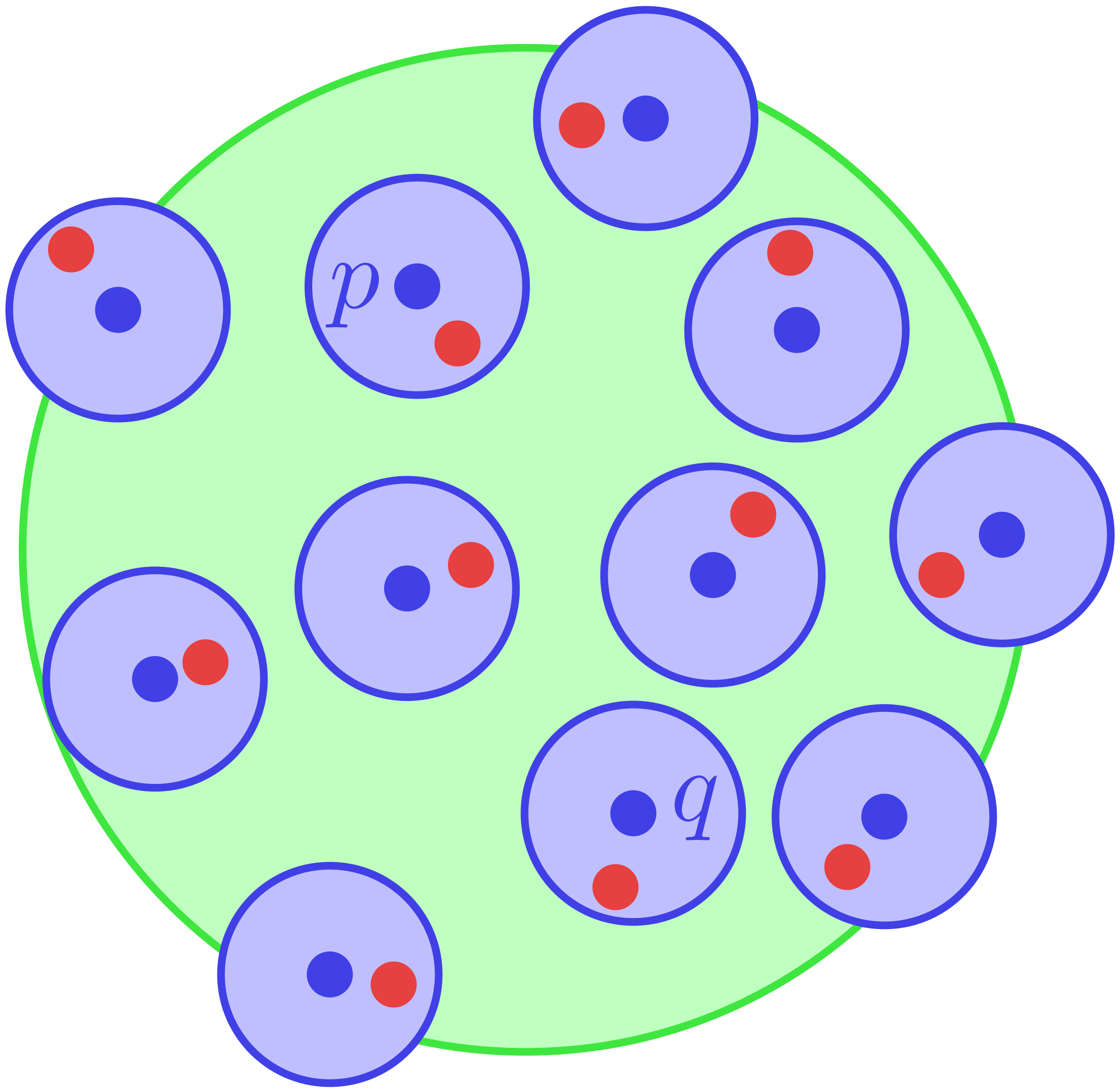
We reconstruct connectivity of the MST edges of P by length (inc

If an edge pq is *short*, the number of other points in C^+_{pq} is constant.



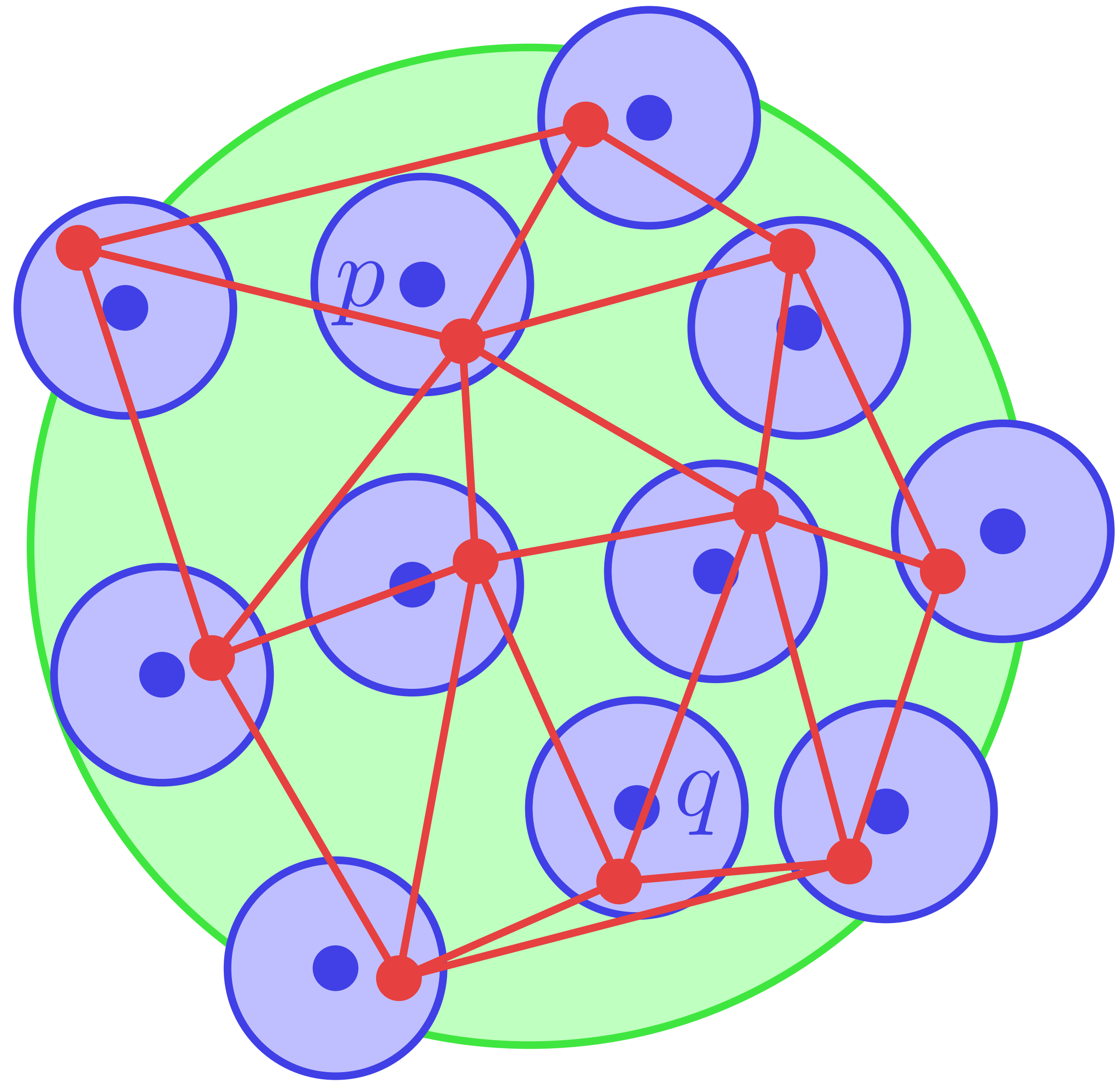
We reconstruct connectivity of the MST edges of P by length (increa...

If an edge pq is *short*, the number of other points in C^+_{pq} is constant.



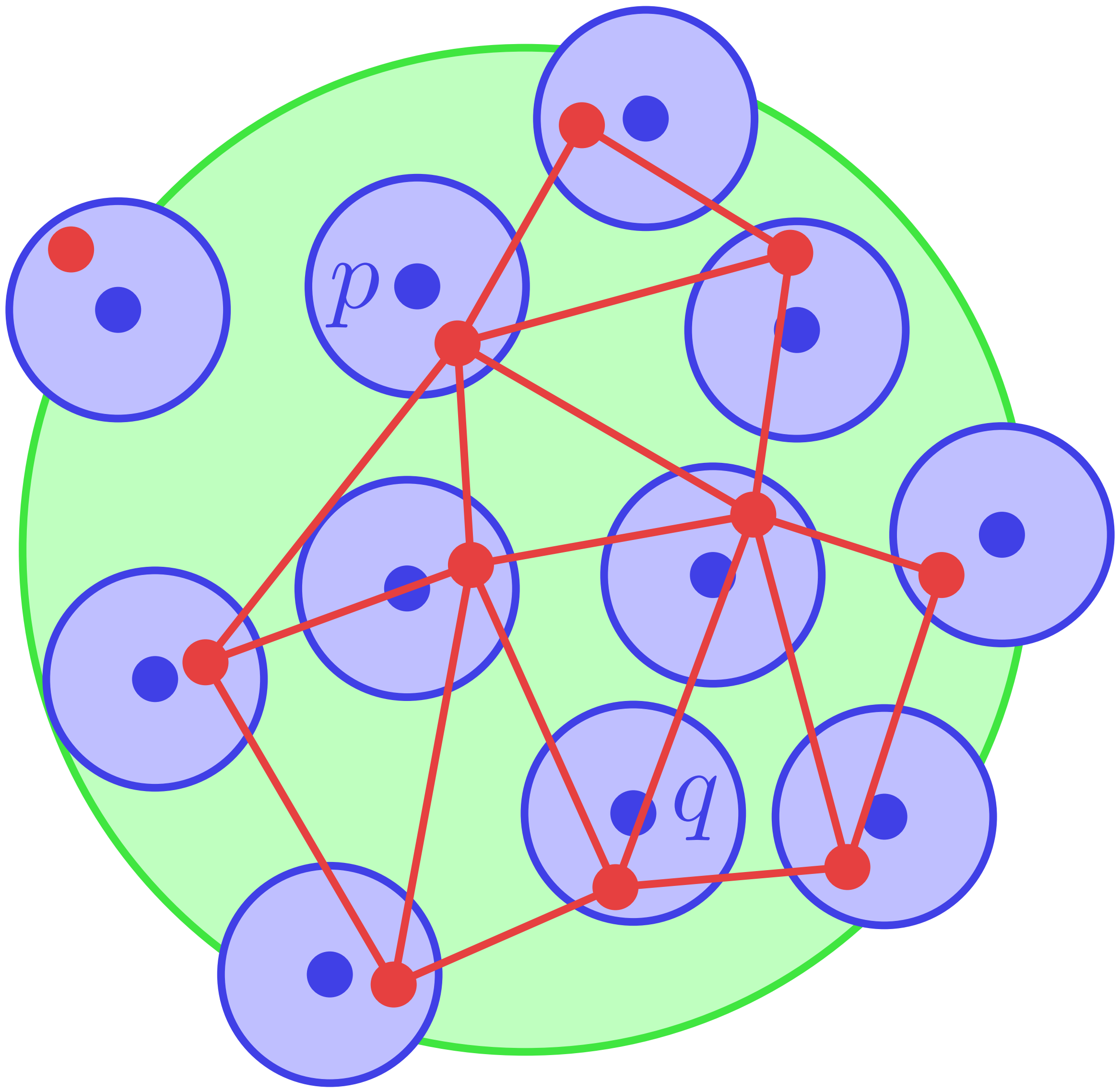
We reconstruct connectivity of the MST edges of P by length (incr

If an edge pq is *short*, the number of other points in C^+_{pq} is constant.



We reconstruct connectivity of the MST edges of P by length (incr...

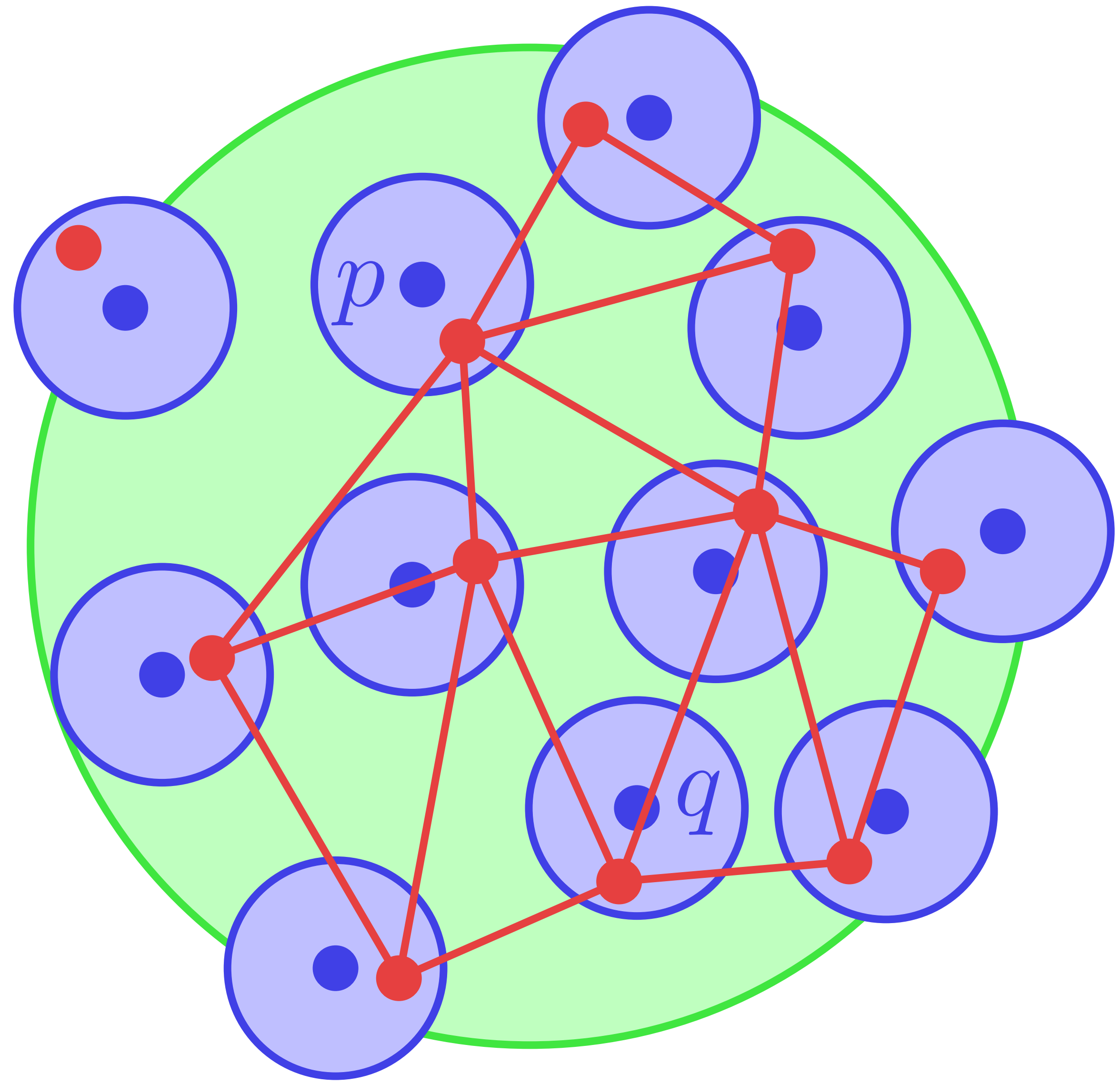
If an edge pq is *short*, the number of other points in C^+_{pq} is constant.



We reconstruct connectivity of the MST edges of P by length (incr

If an edge pq is *short*, the number of other points in

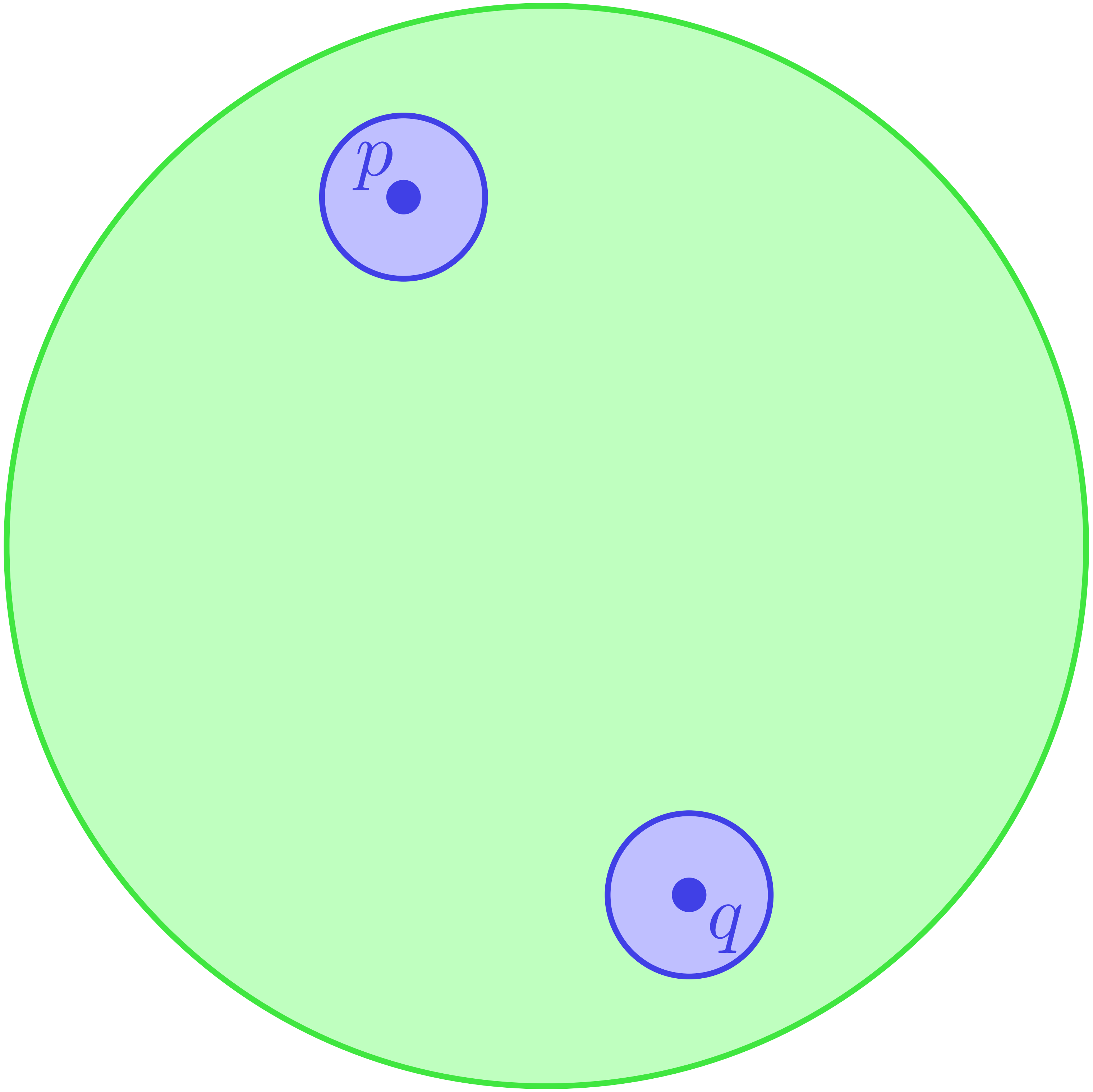
If an edge pq is *long*, all other points in C_{pq}^{+} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (increa…

If an edge pq is *short*, the number of other points in …

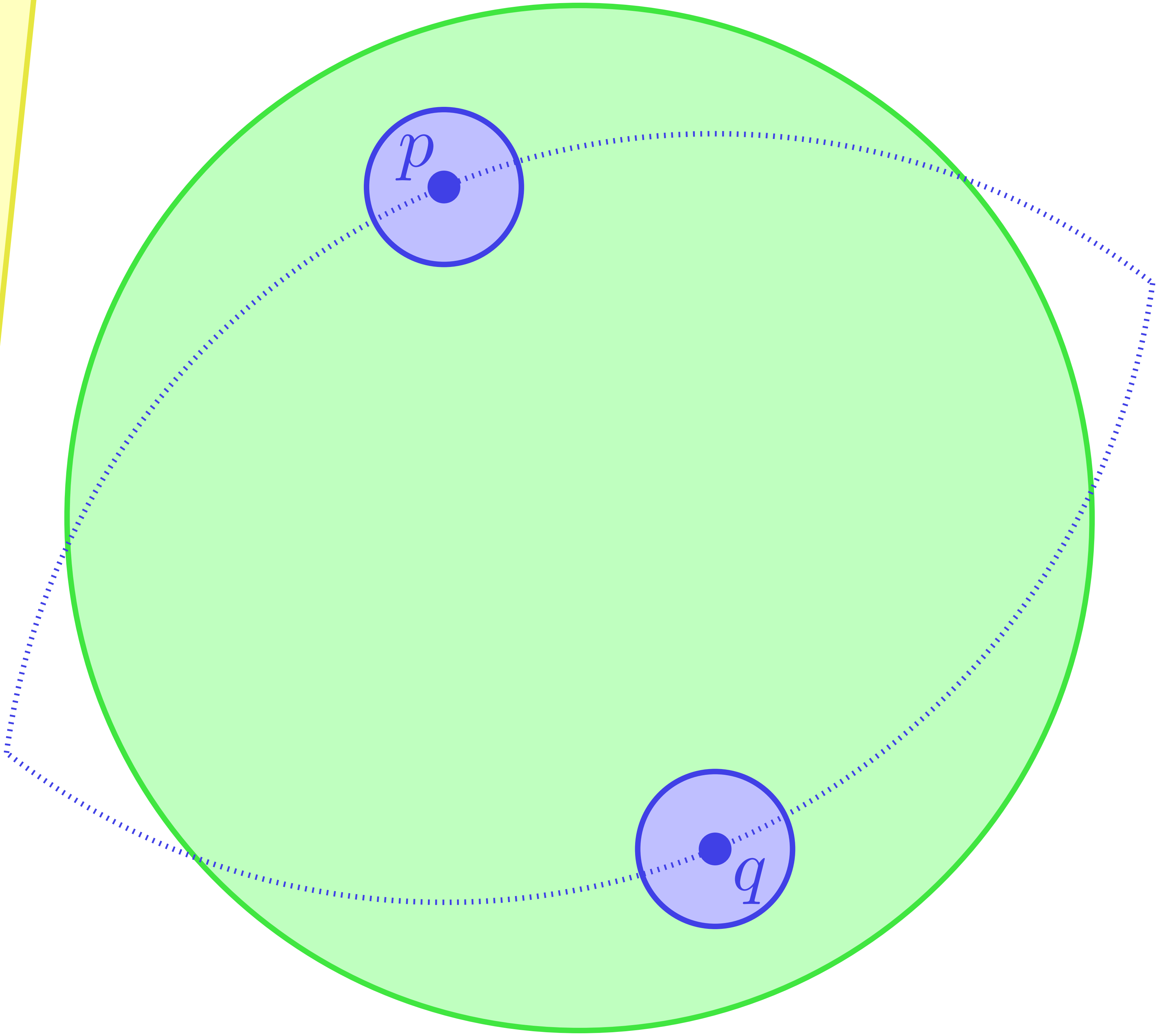
If an edge pq is *long*, all other points in C^+_{pq} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incr

If an edge pq is *short*, the number of other points in

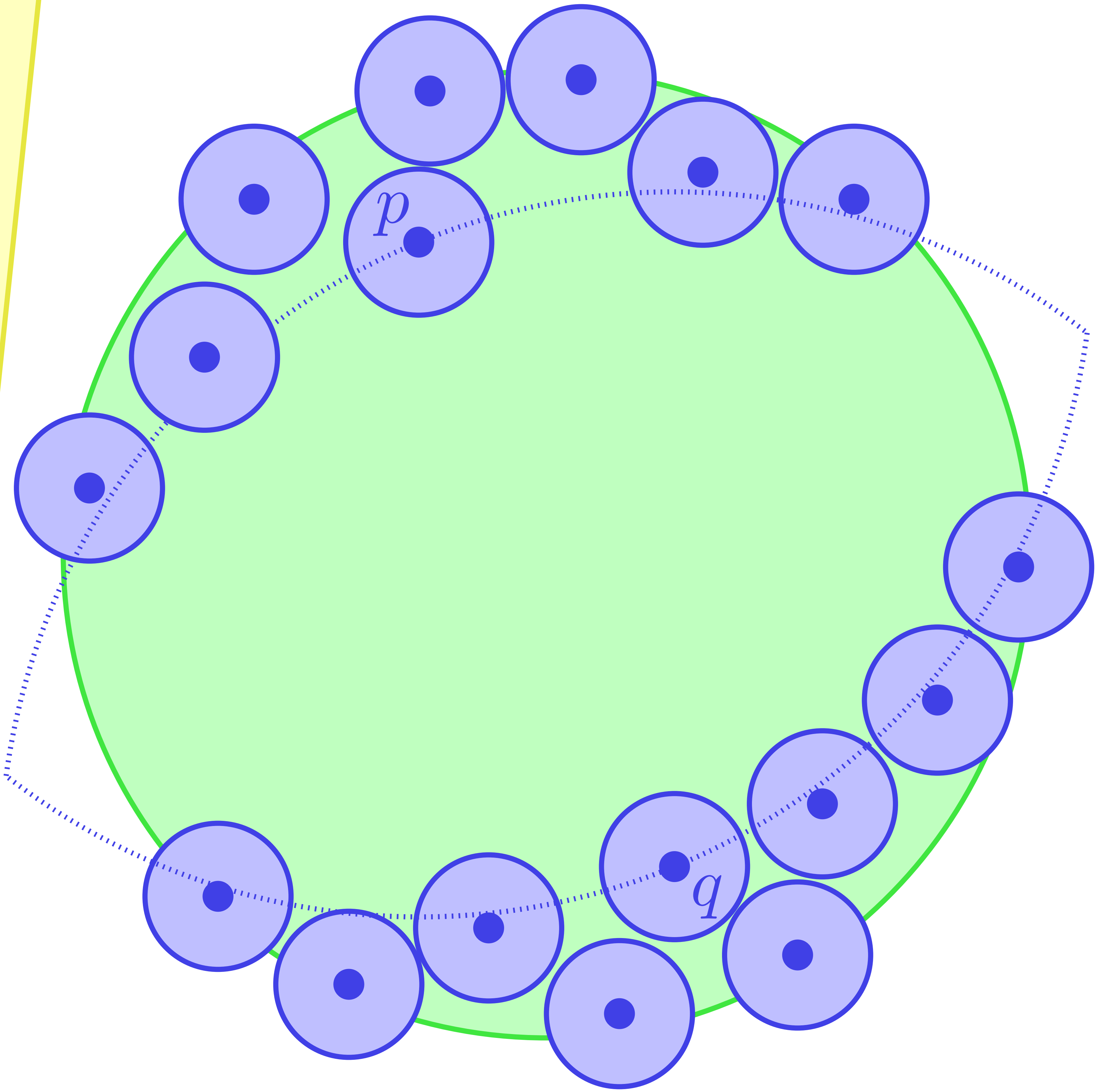
If an edge pq is *long*, all other points in C^+_{pq} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incr

If an edge pq is *short*, the number of other points in

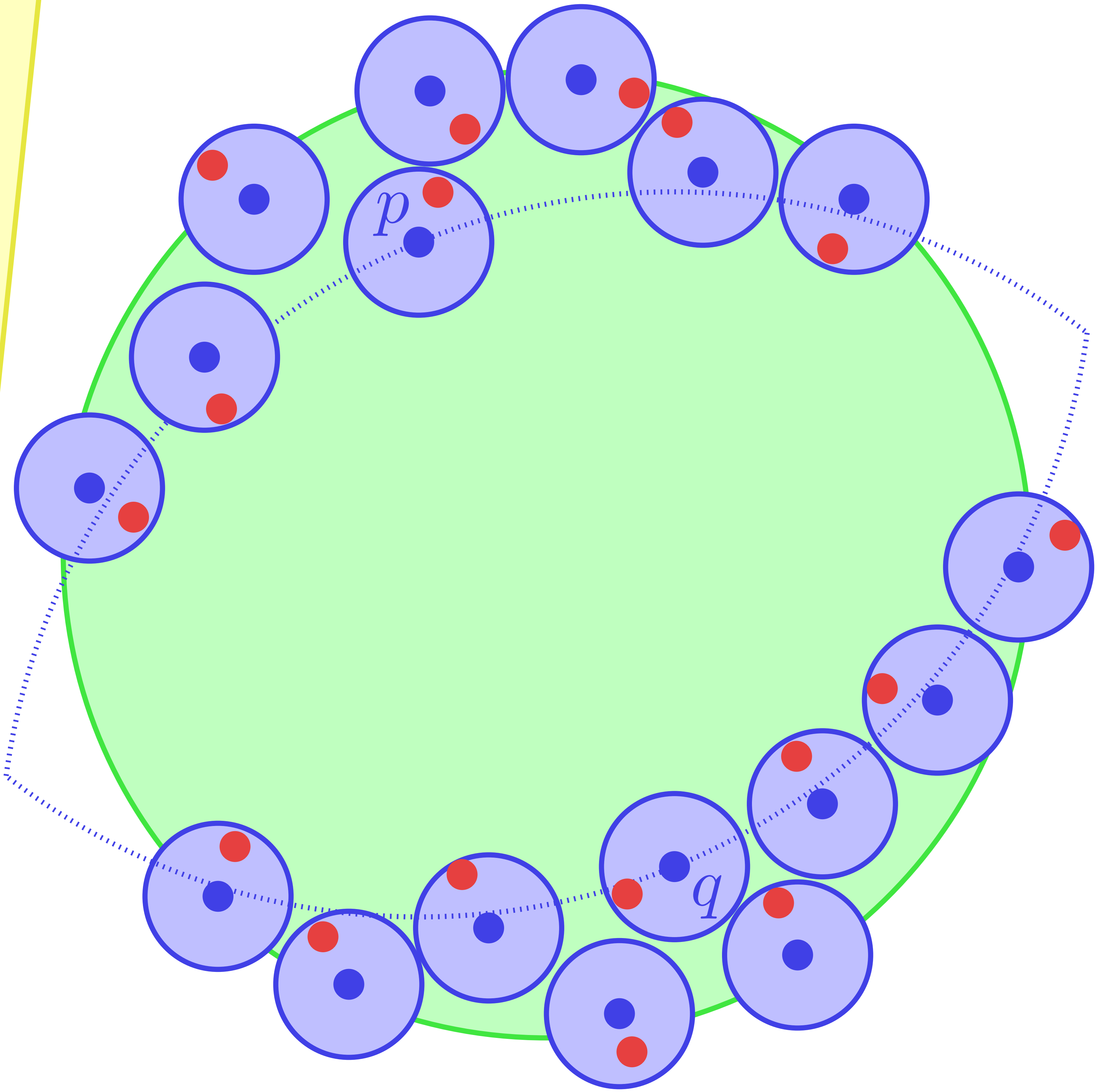
If an edge pq is *long*, all other points in C_{pq}^{+} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incr

If an edge pq is *short*, the number of other points in

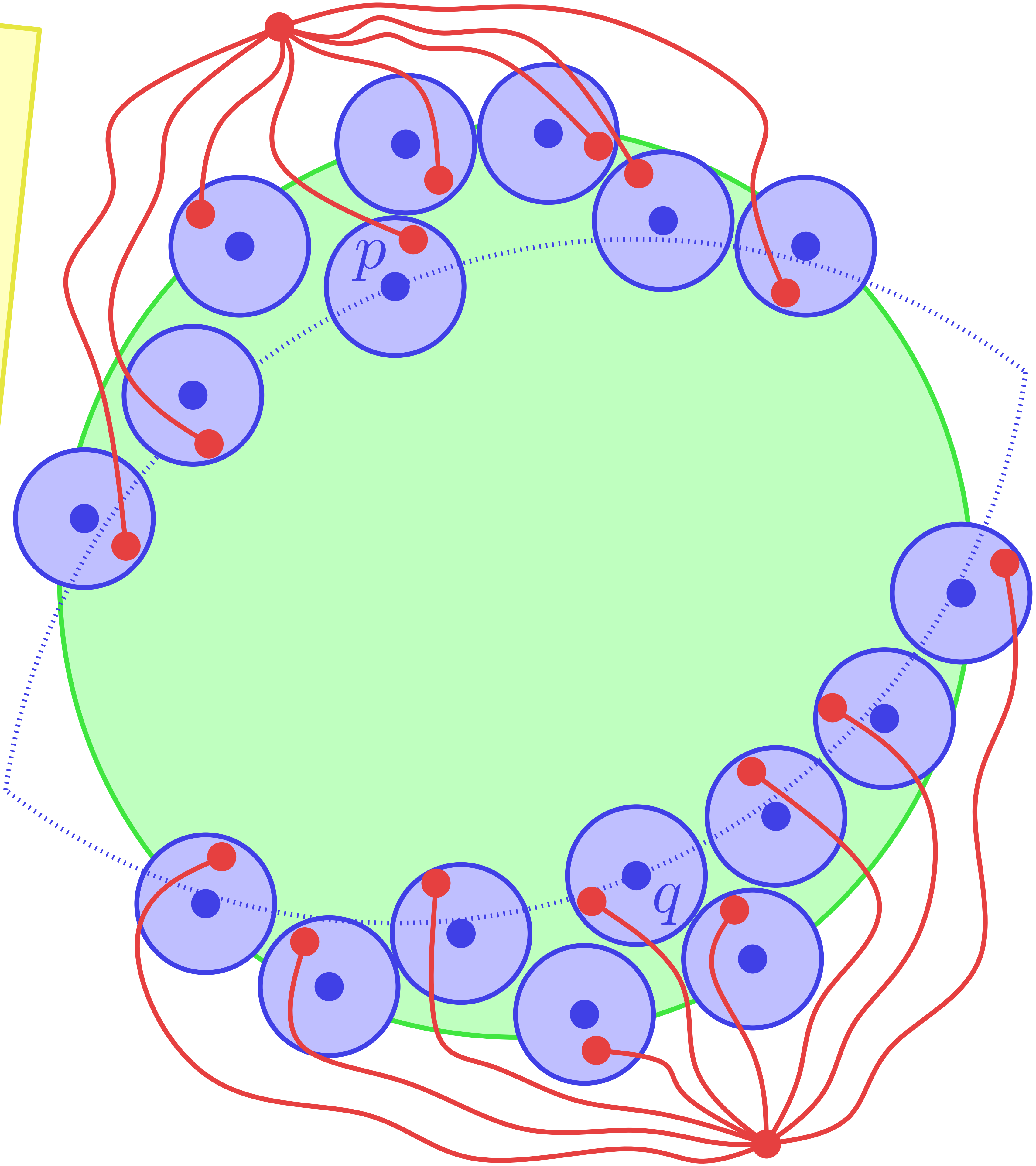
If an edge pq is *long*, all other points in C^+_{pq} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incr

If an edge pq is *short*, the number of other points in

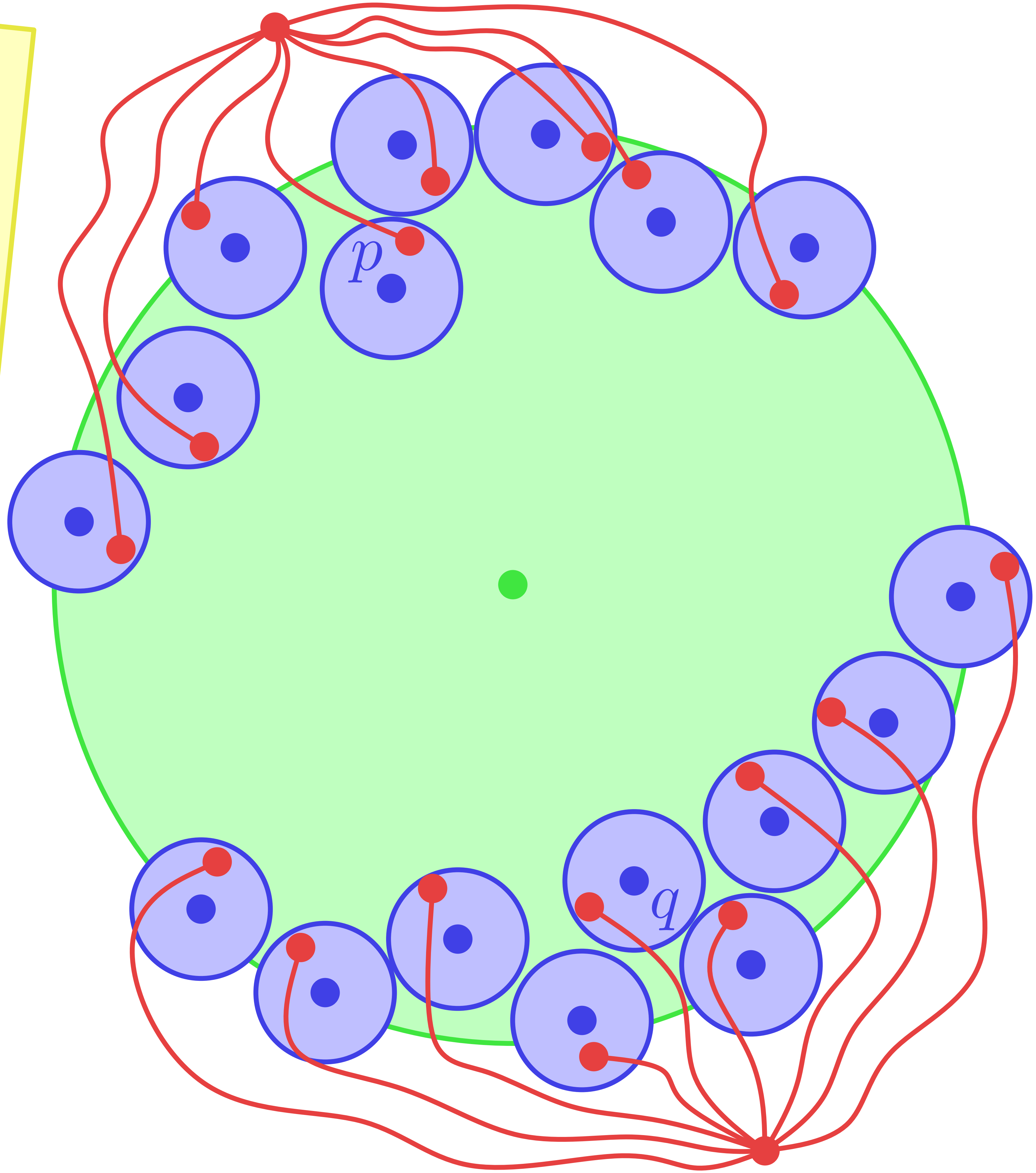
If an edge pq is *long*, all other points in C_{pq}^{+} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incre

If an edge pq is *short*, the number of other points in

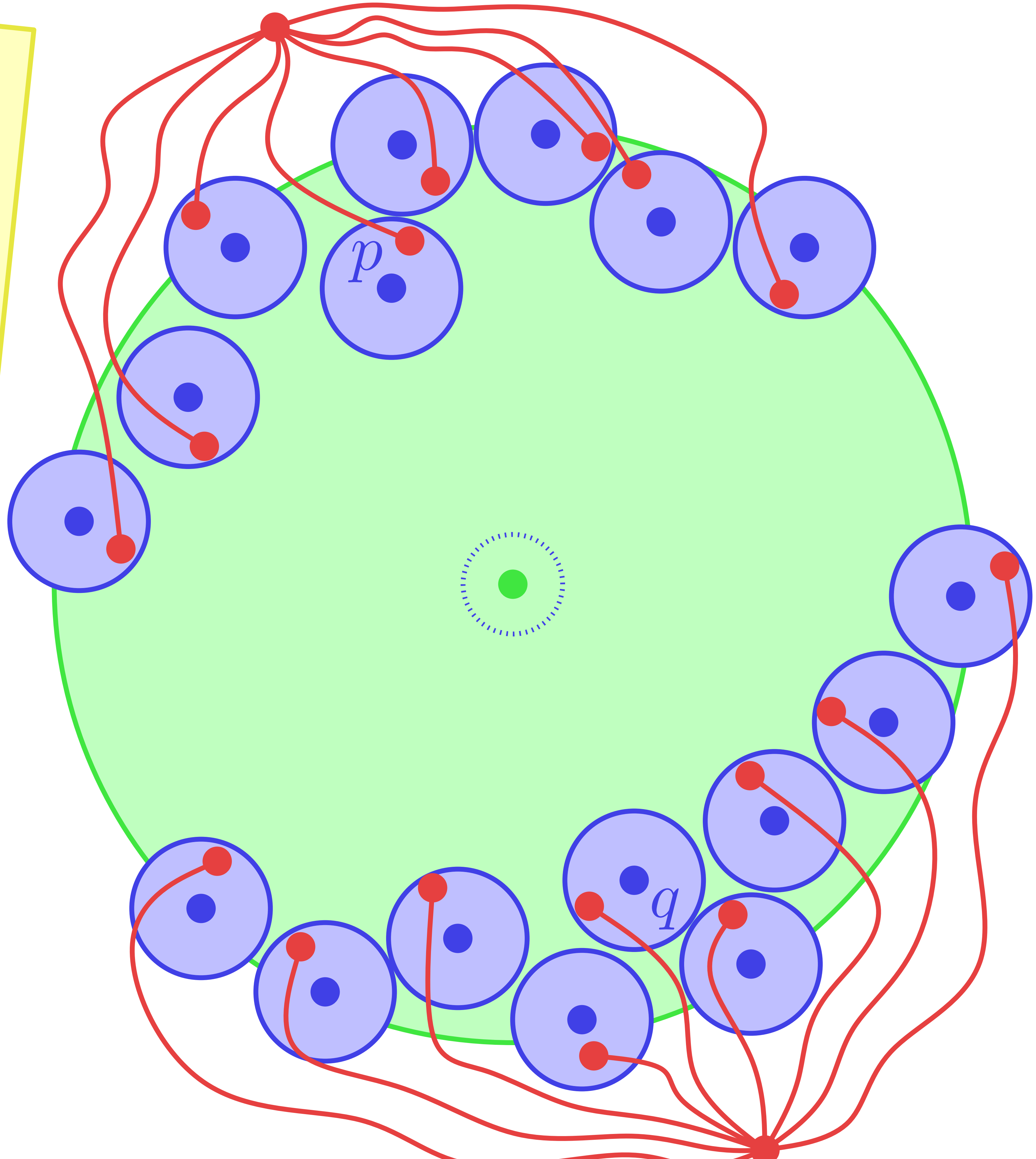
If an edge pq is *long*, all other points in C_{pq}^{+} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incre

If an edge pq is *short*, the number of other points in

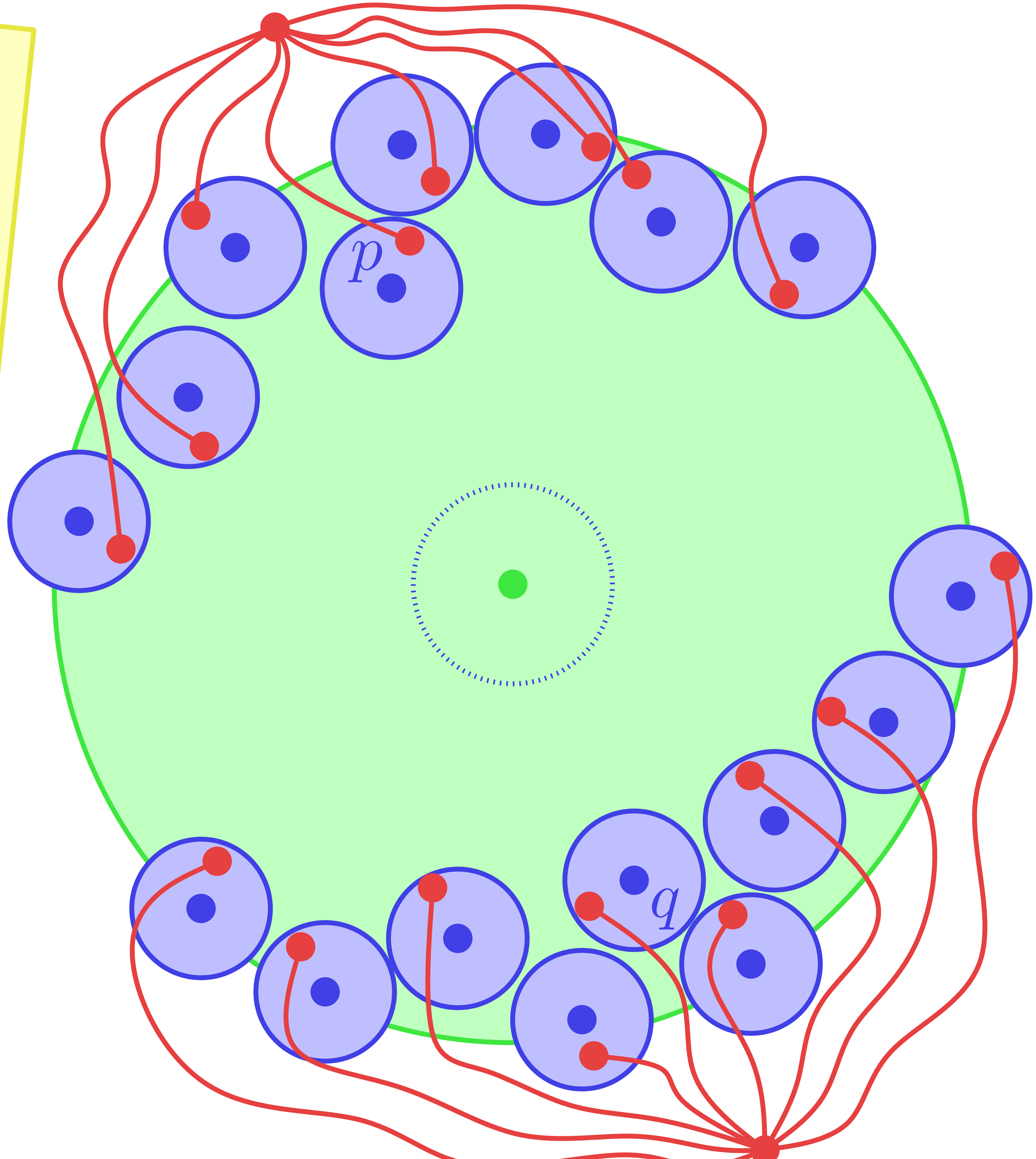
If an edge pq is *long*, all other points in C^+_{pq} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incr

If an edge pq is *short*, the number of other points in

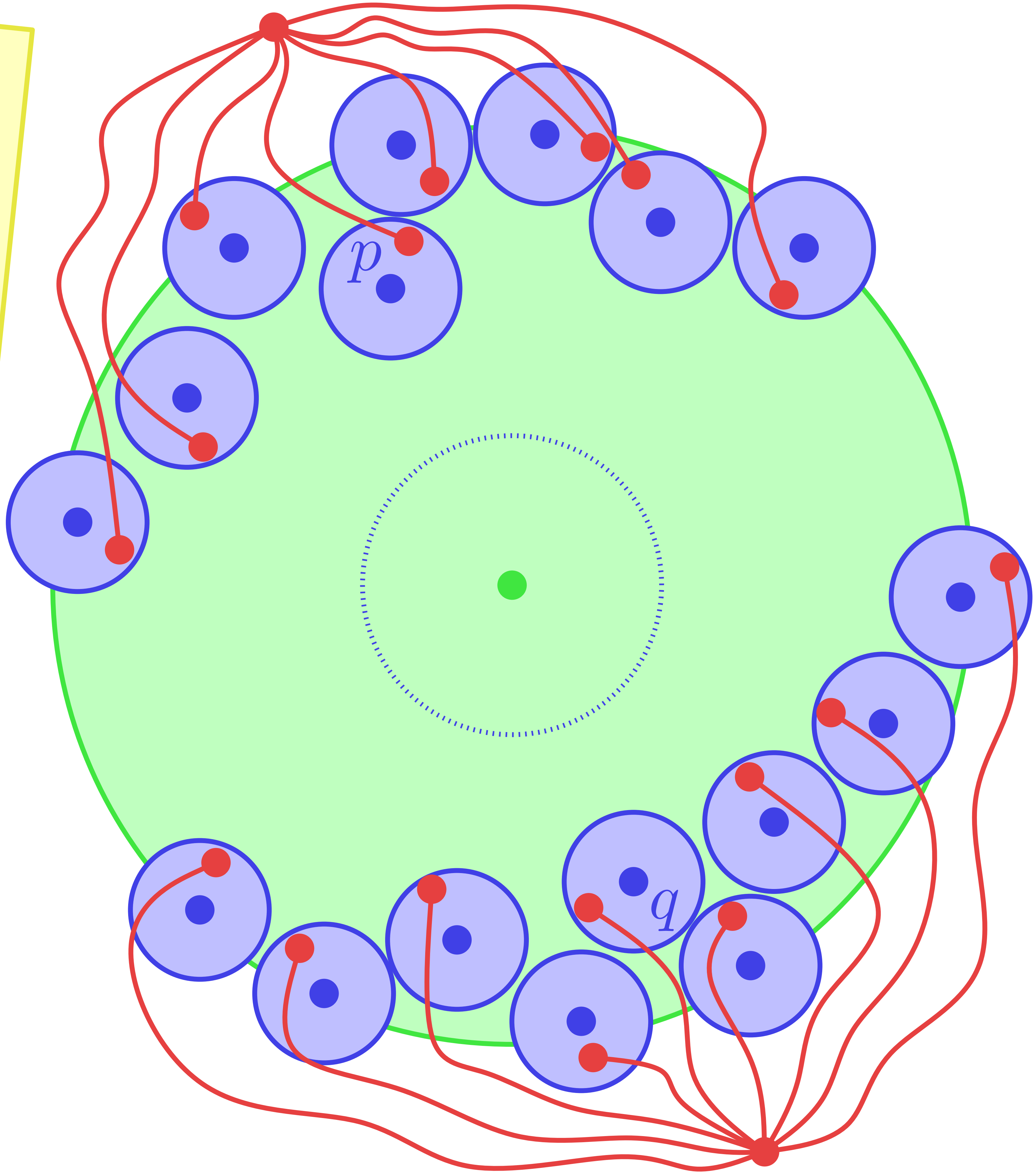
If an edge pq is *long*, all other points in C^+_{pq} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incre

If an edge pq is *short*, the number of other points in

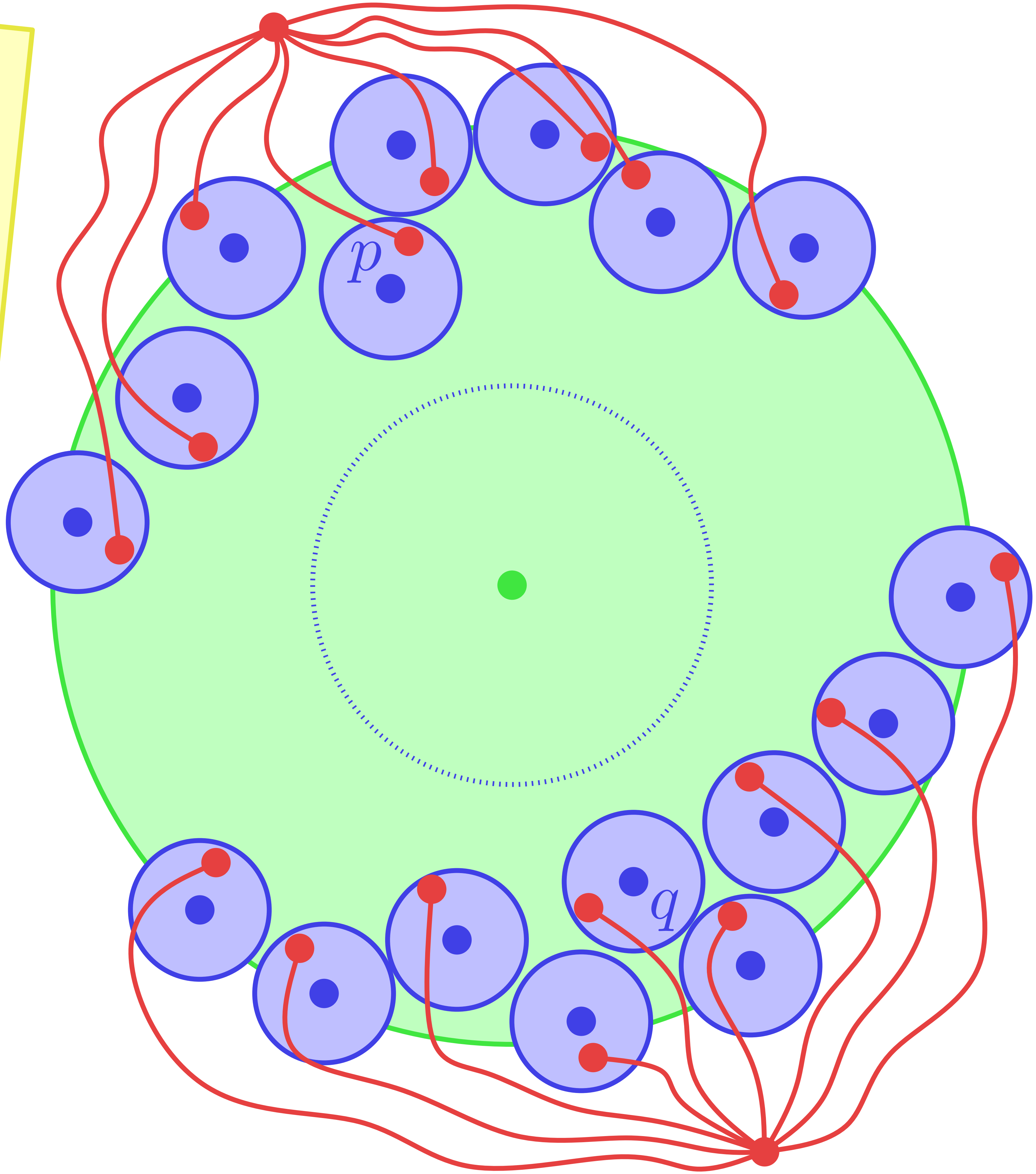
If an edge pq is *long*, all other points in C_{pq}^{+} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incr

If an edge pq is *short*, the number of other points in

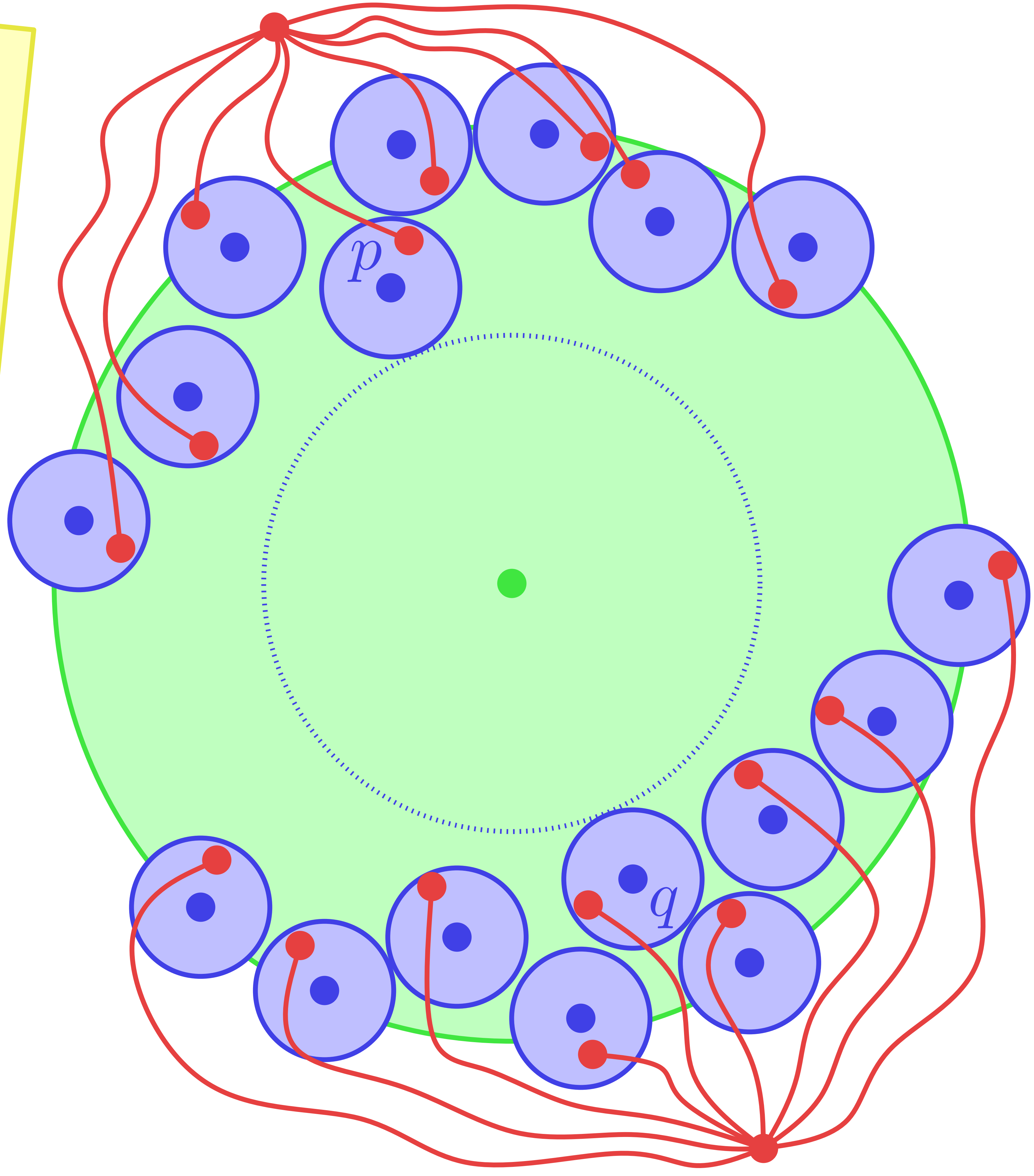
If an edge pq is *long*, all other points in C^+_{pq} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incre…

If an edge pq is *short*, the number of other points in …

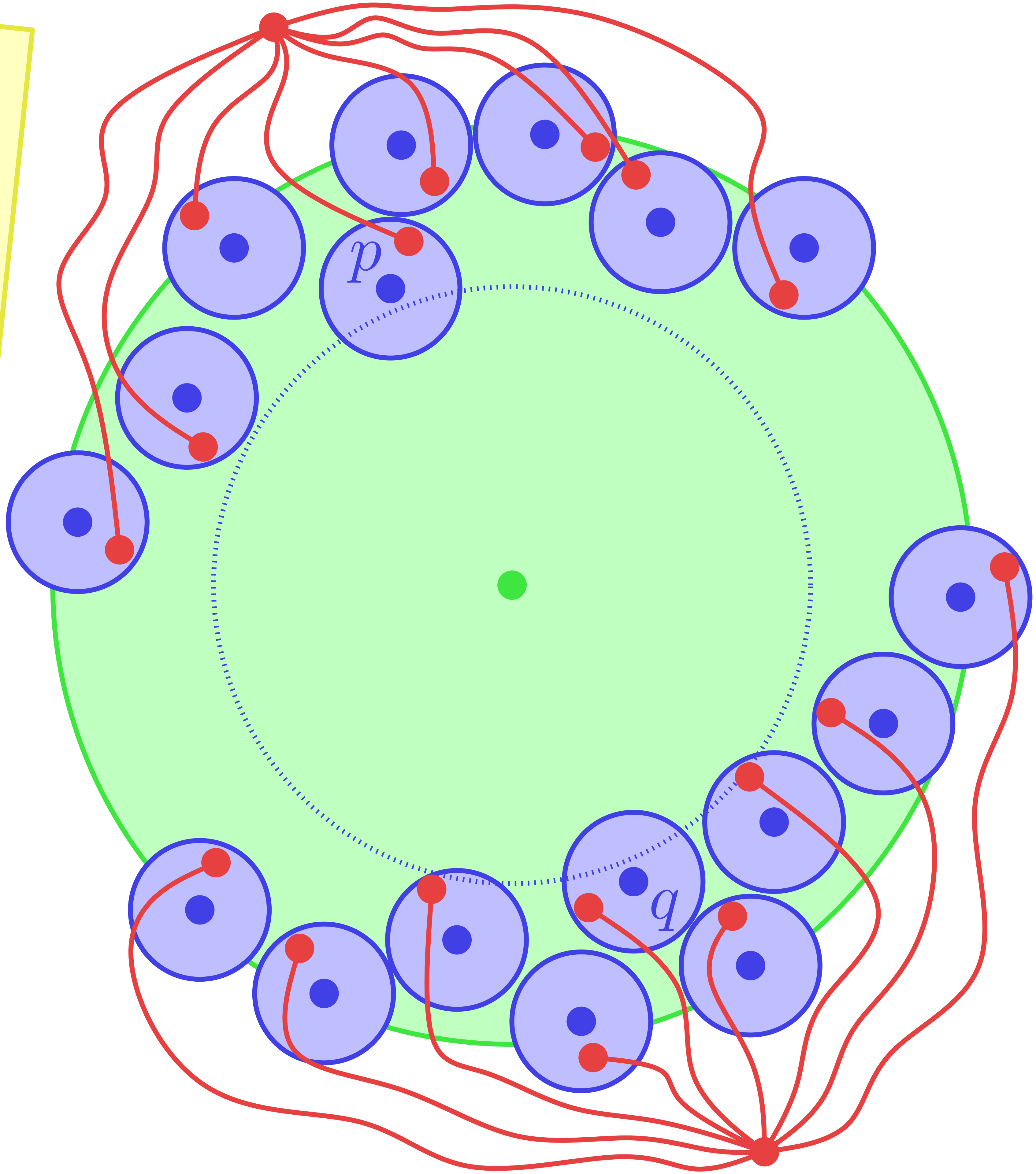
If an edge pq is *long*, all other points in C^+_{pq} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incr

If an edge pq is *short*, the number of other points in

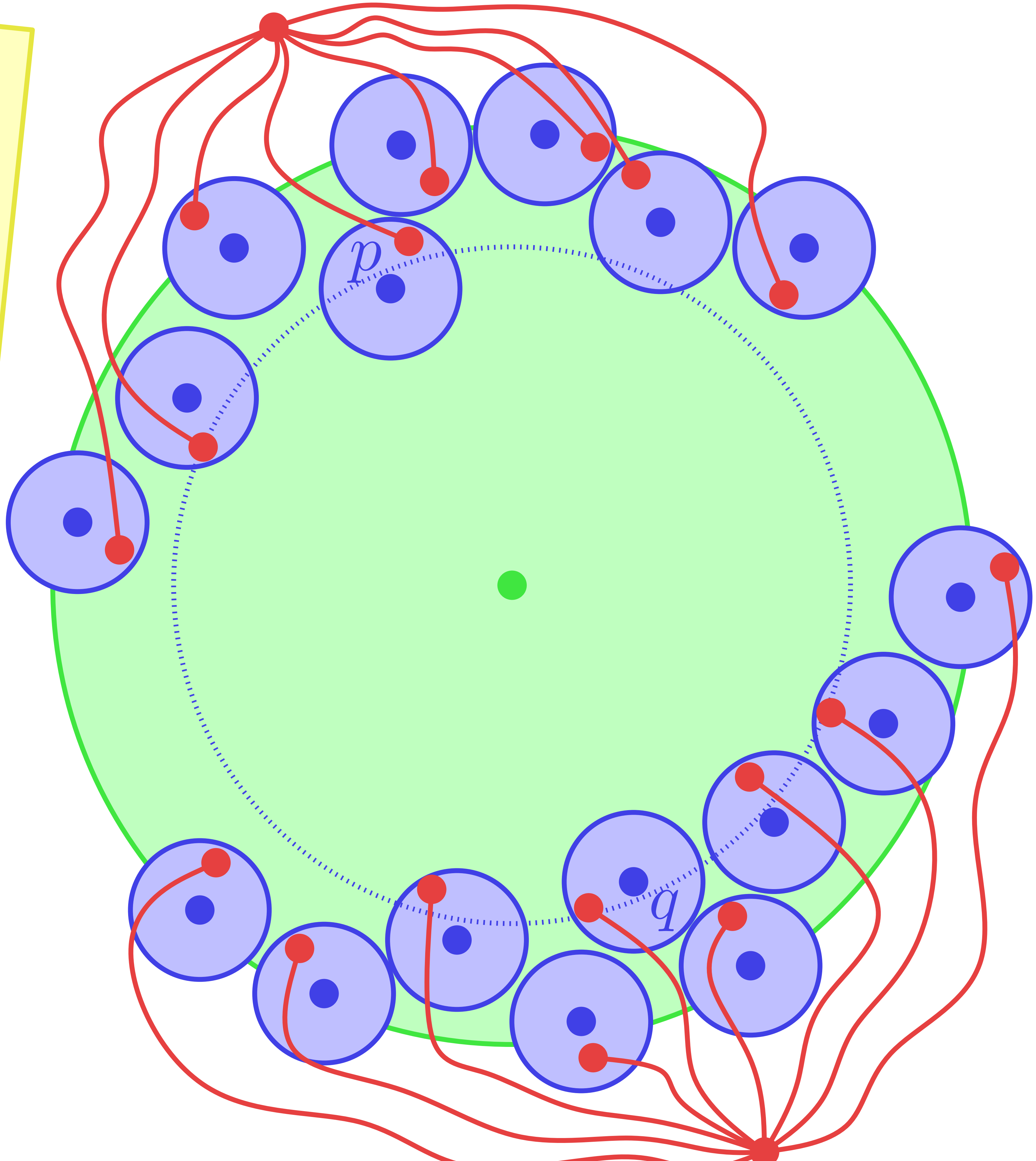
If an edge pq is *long*, all other points in C^+_{pq} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incr

If an edge pq is *short*, the number of other points in

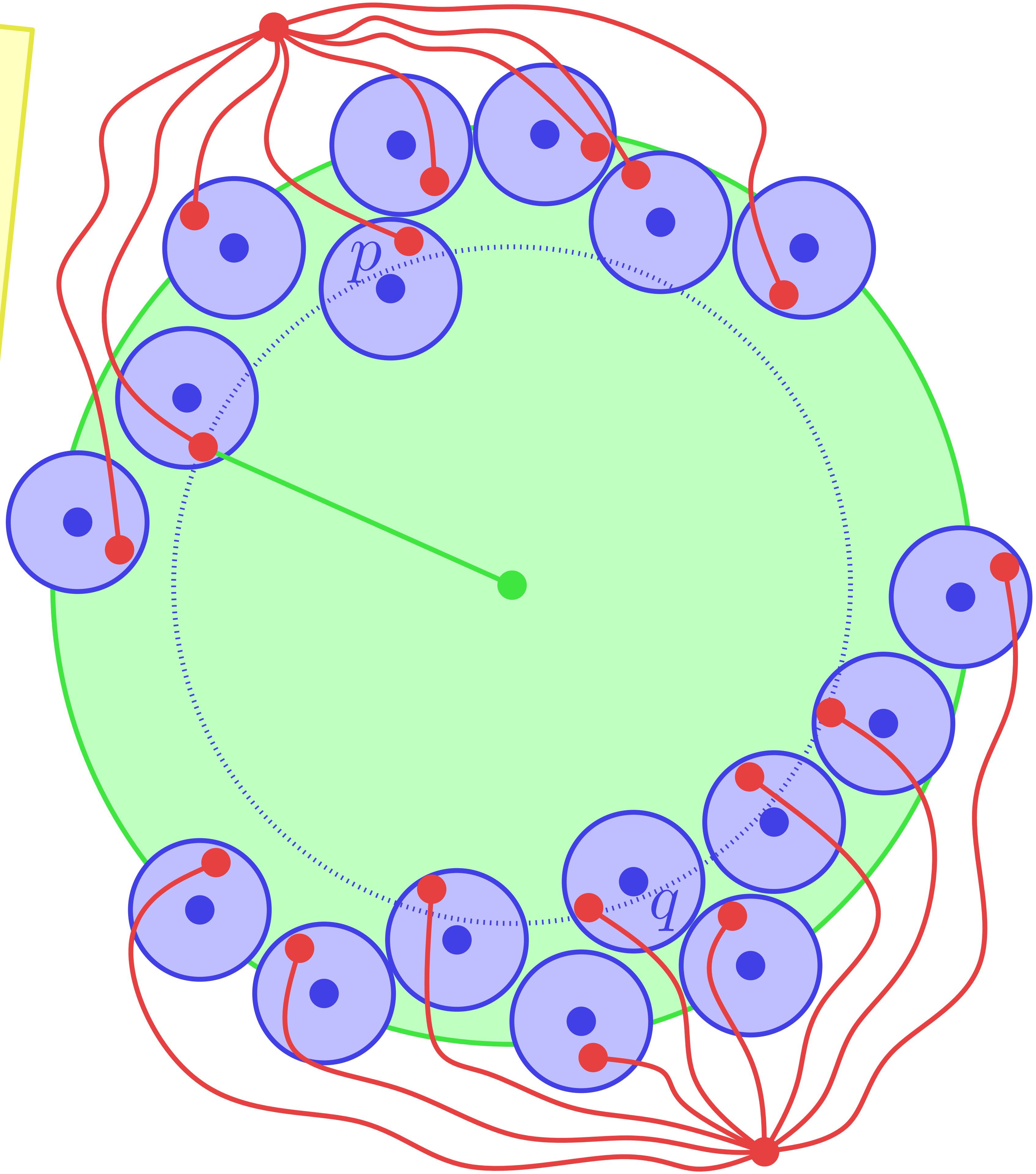
If an edge pq is *long*, all other points in C_{pq}^{+} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incr

If an edge pq is *short*, the number of other points in

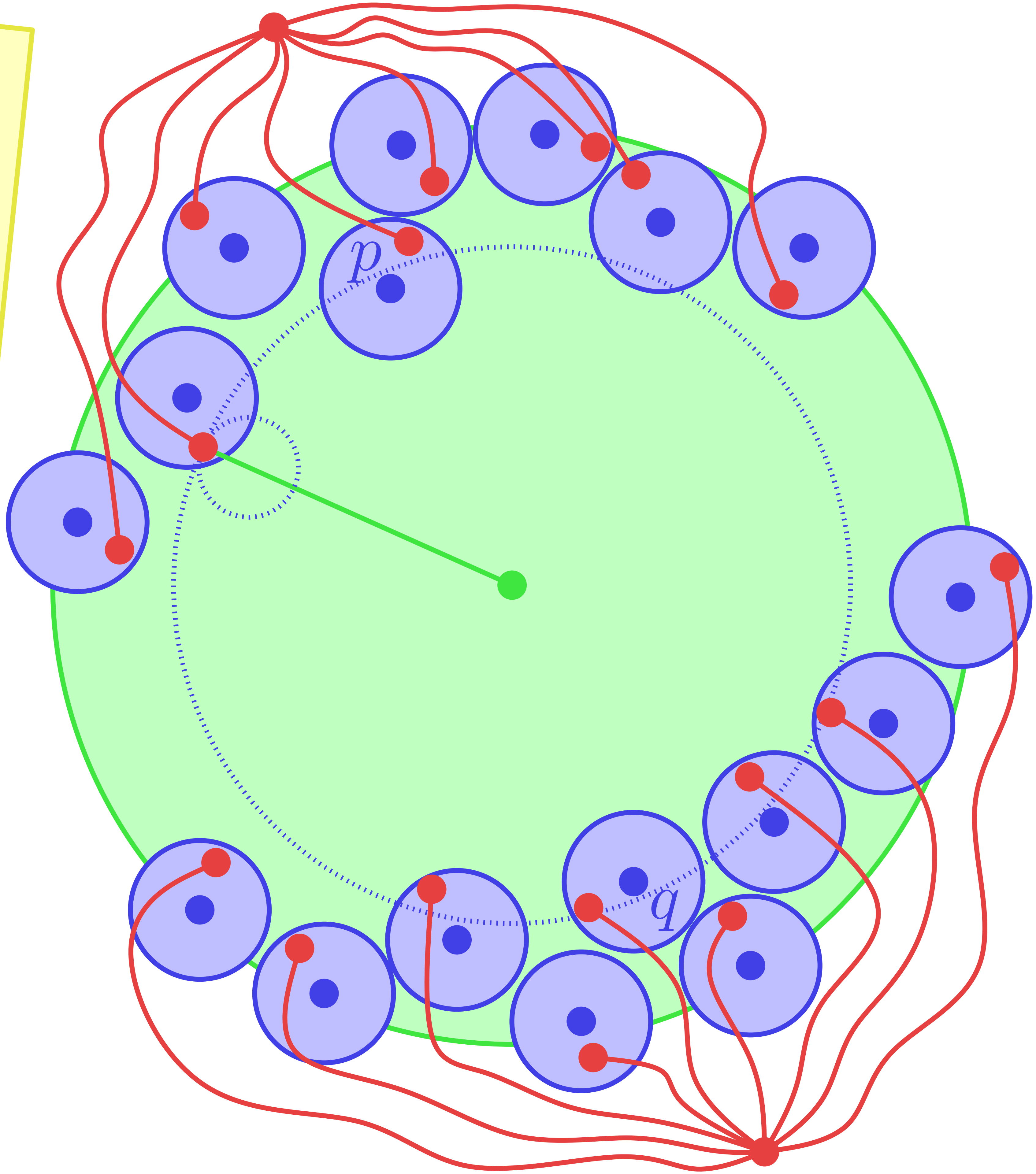
If an edge pq is *long*, all other points in C_{pq}^{+} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incr

If an edge pq is *short*, the number of other points in

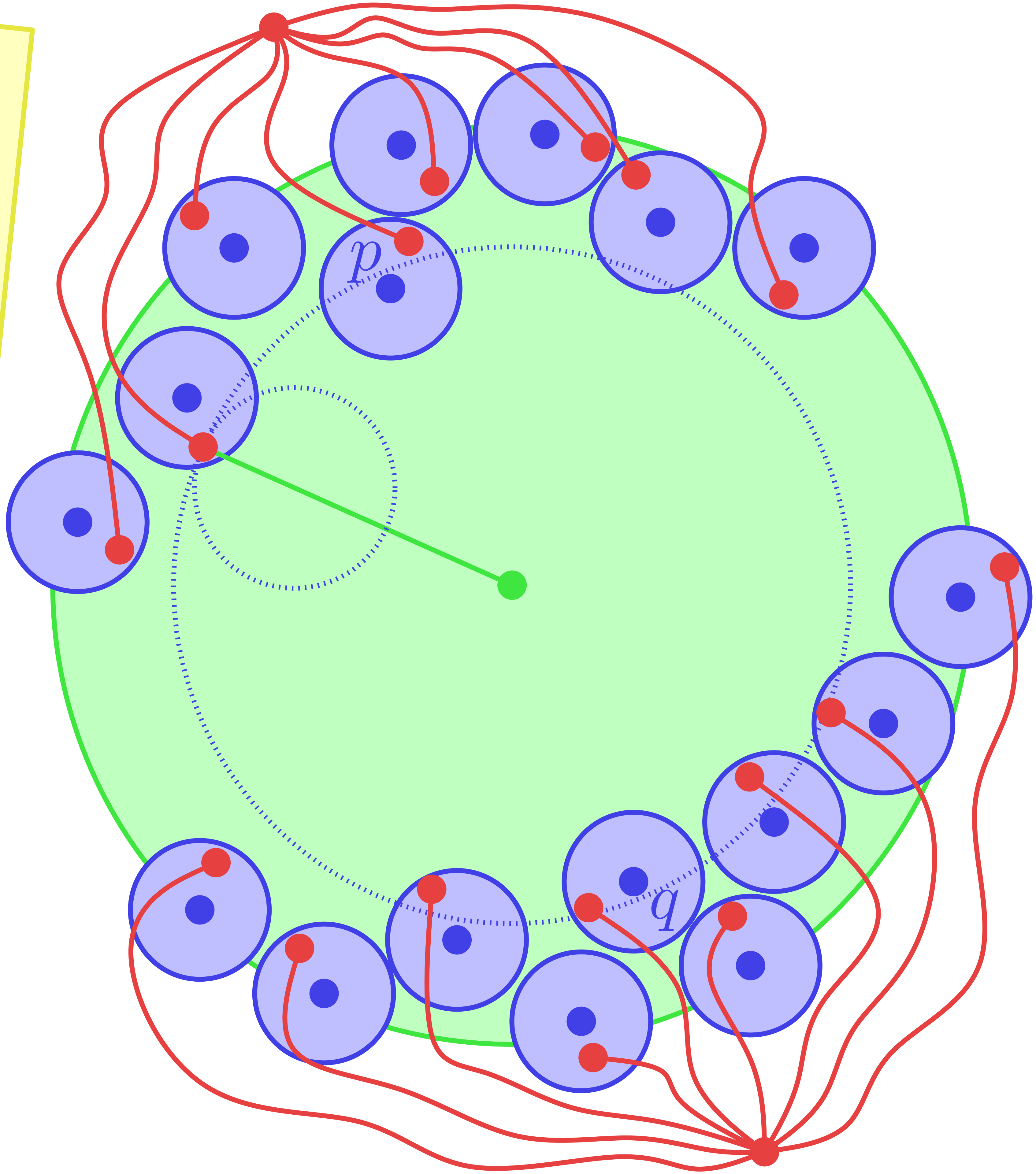
If an edge pq is *long*, all other points in C_{pq}^{+} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incre

If an edge pq is *short*, the number of other points in

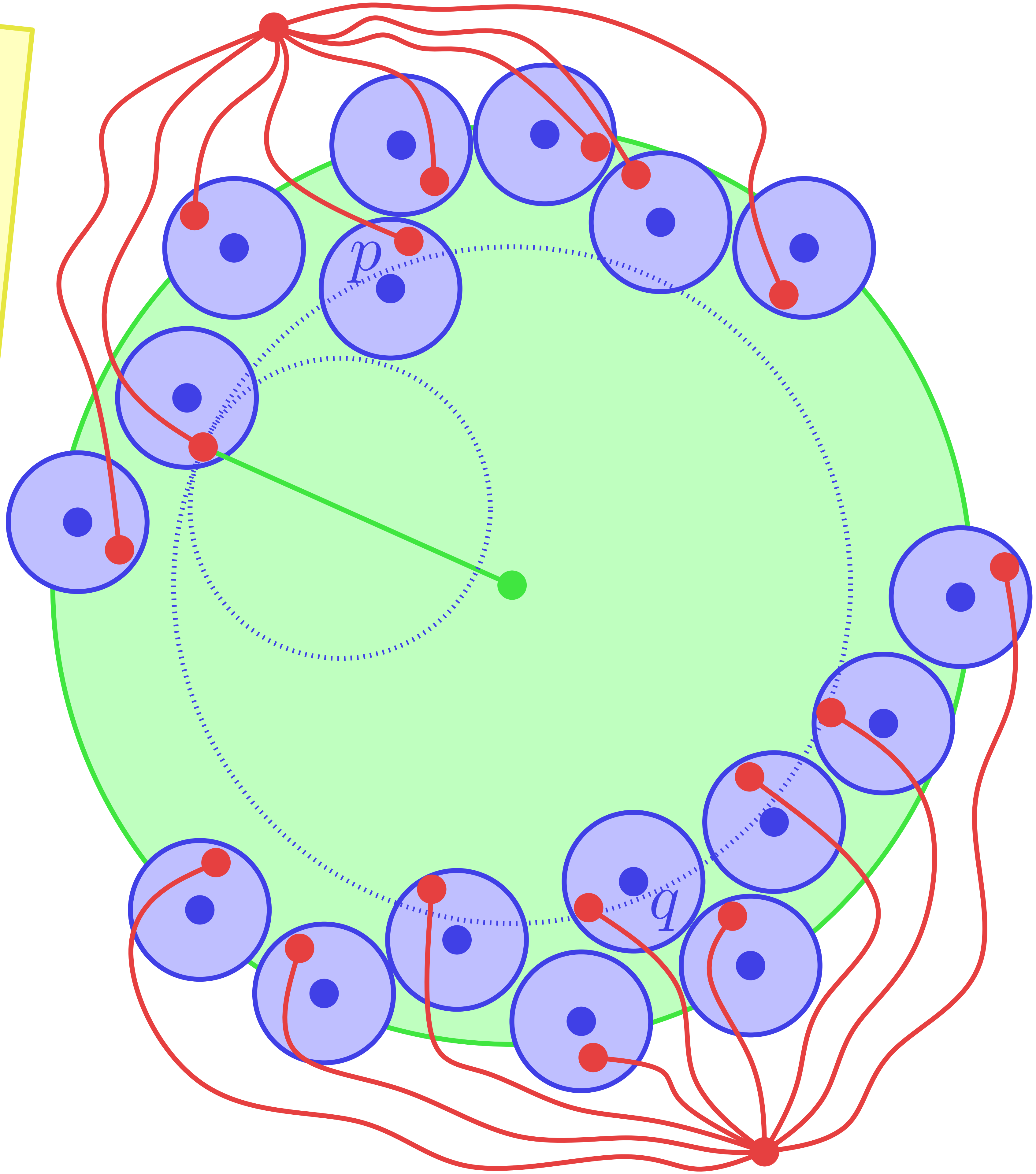
If an edge pq is *long*, all other points in C_{pq}^{+} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incre…

If an edge pq is *short*, the number of other points in …

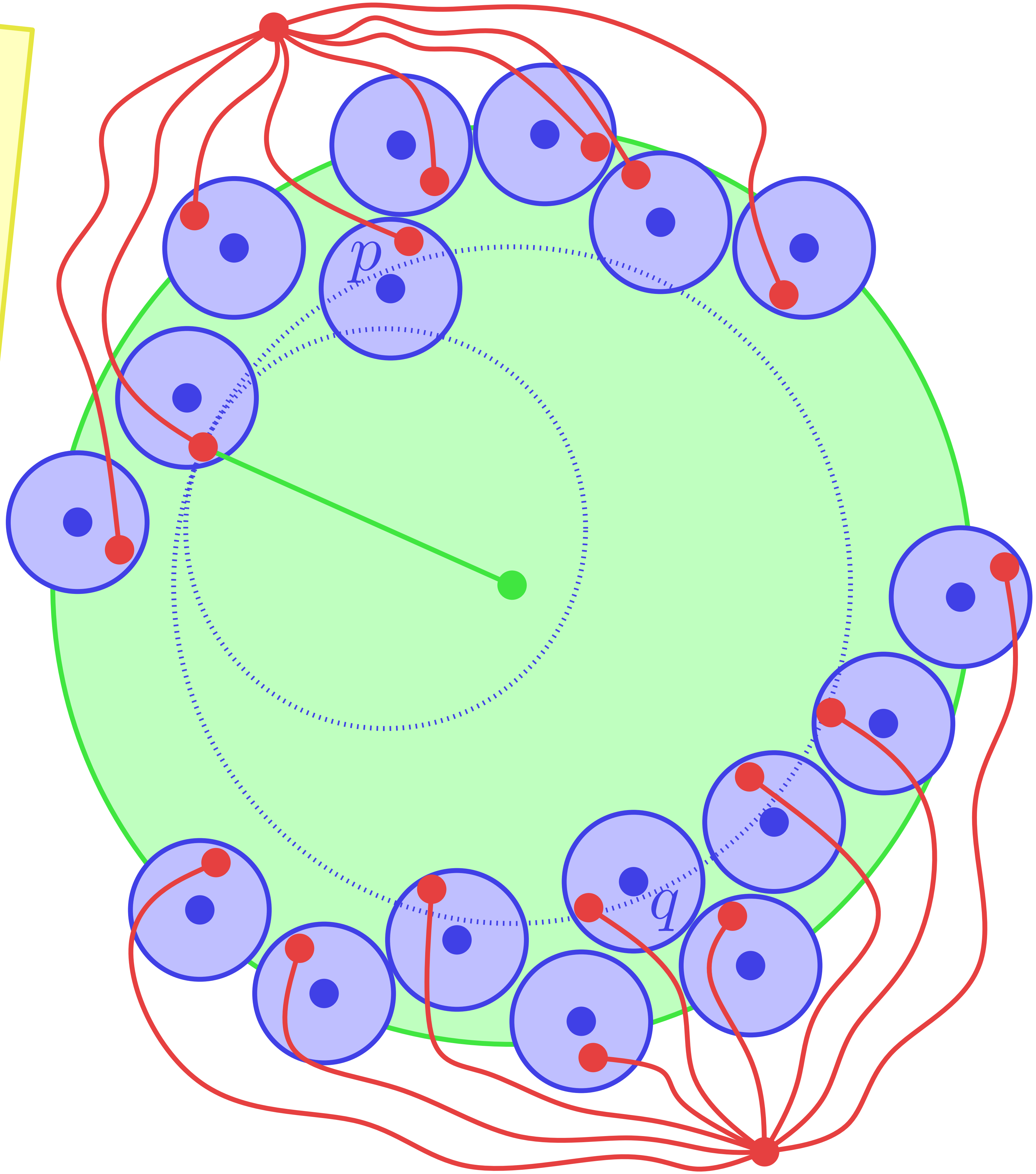
If an edge pq is *long*, all other points in C_{pq}^{+} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incre

If an edge pq is *short*, the number of other points in

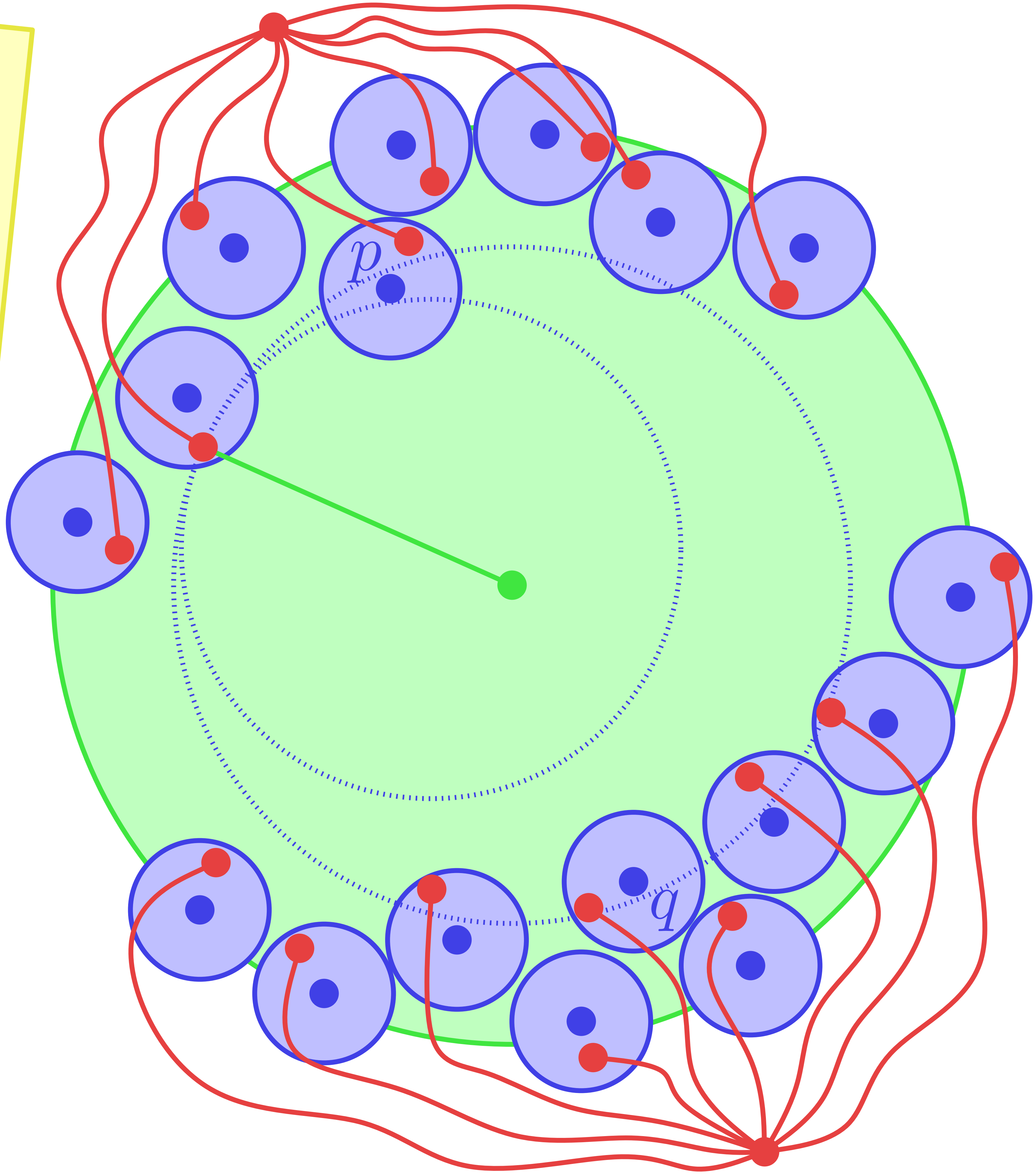
If an edge pq is *long*, all other points in C_{pq}^{+} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incr

If an edge pq is *short*, the number of other points in

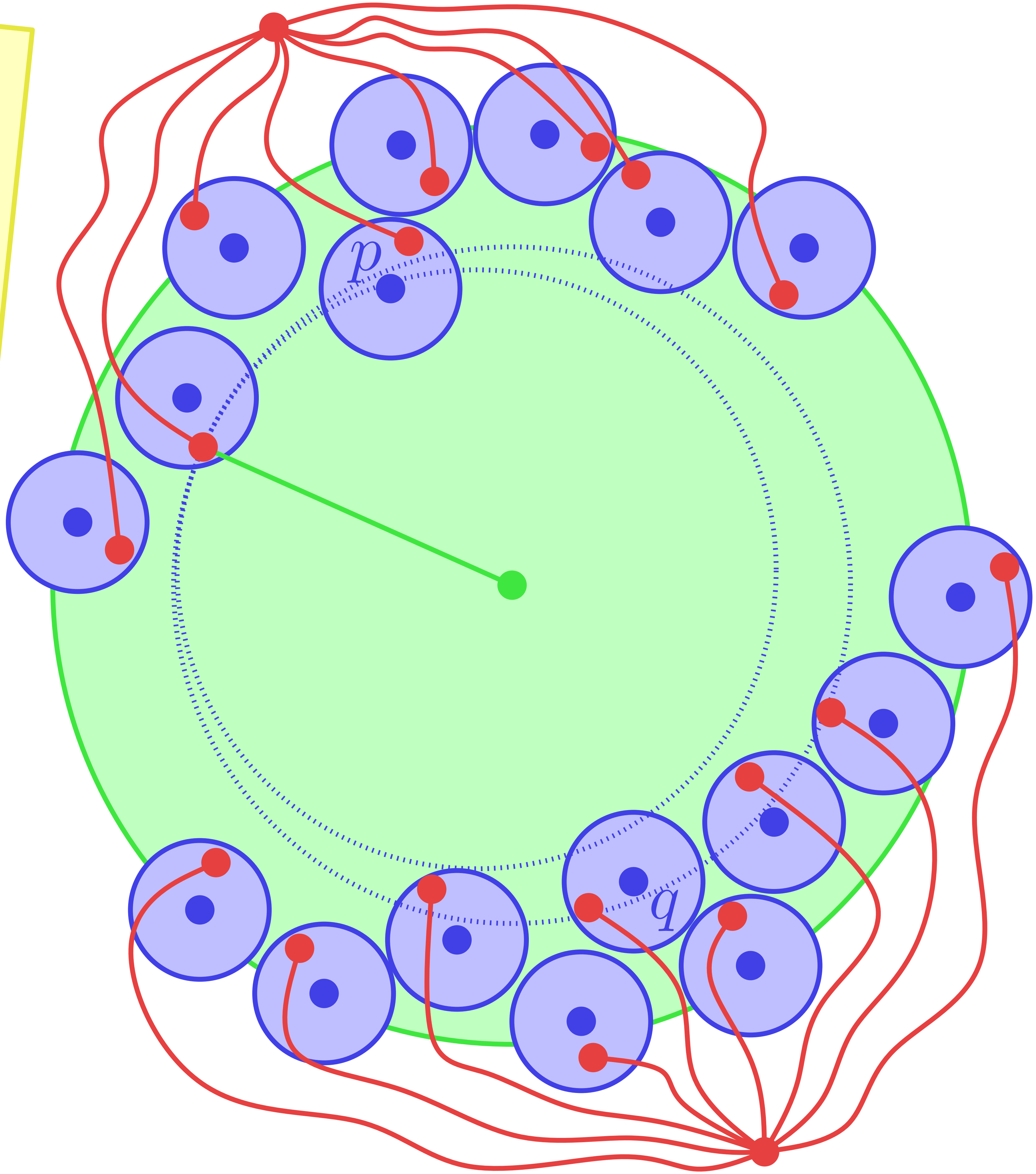
If an edge pq is *long*, all other points in C^+_{pq} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incre

If an edge pq is *short*, the number of other points in

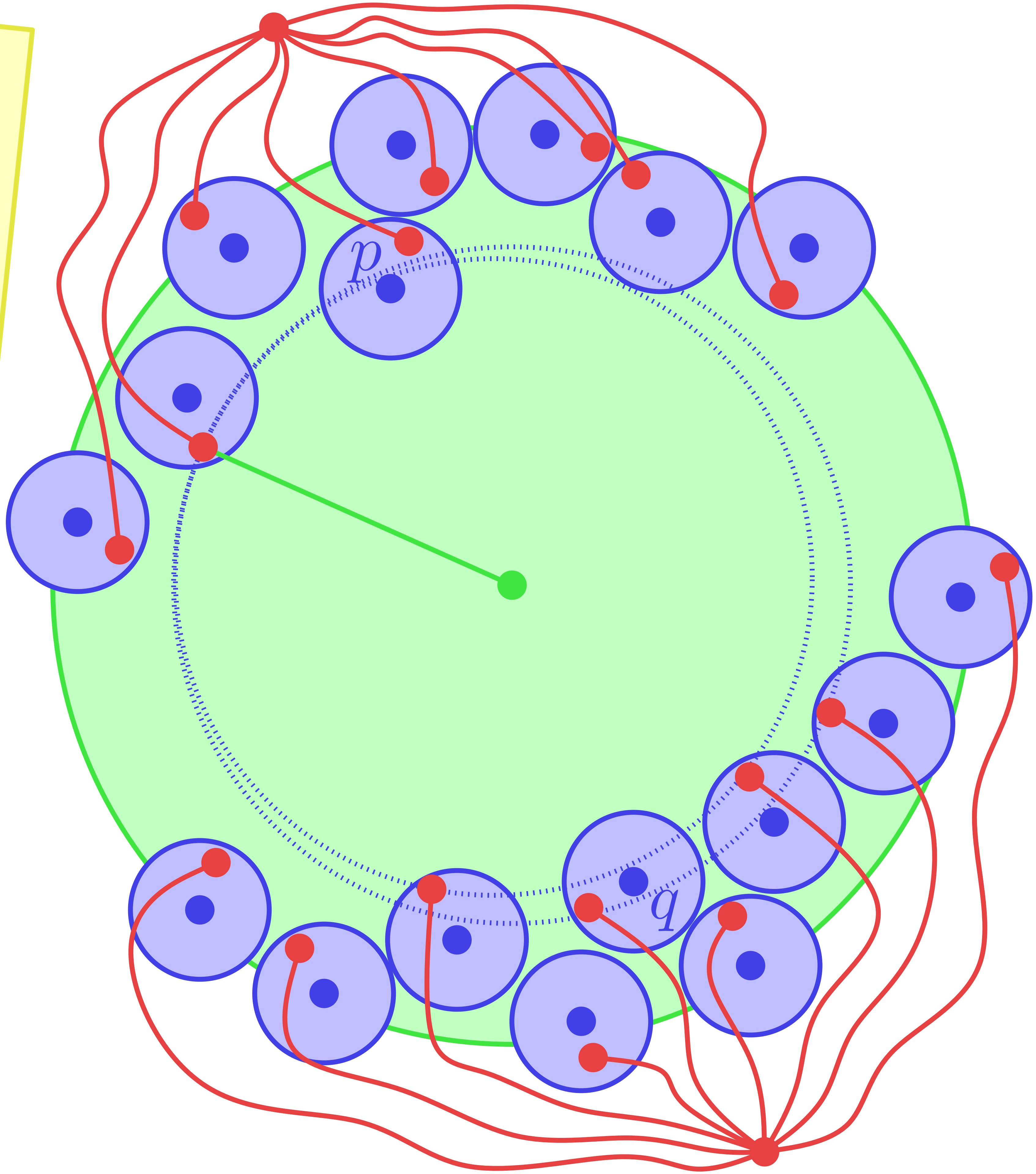
If an edge pq is *long*, all other points in C^+_{pq} are already connected to either p or q.



We reconstruct connectivity of the MST edges of P by length (incr

If an edge pq is *short*, the number of other points in

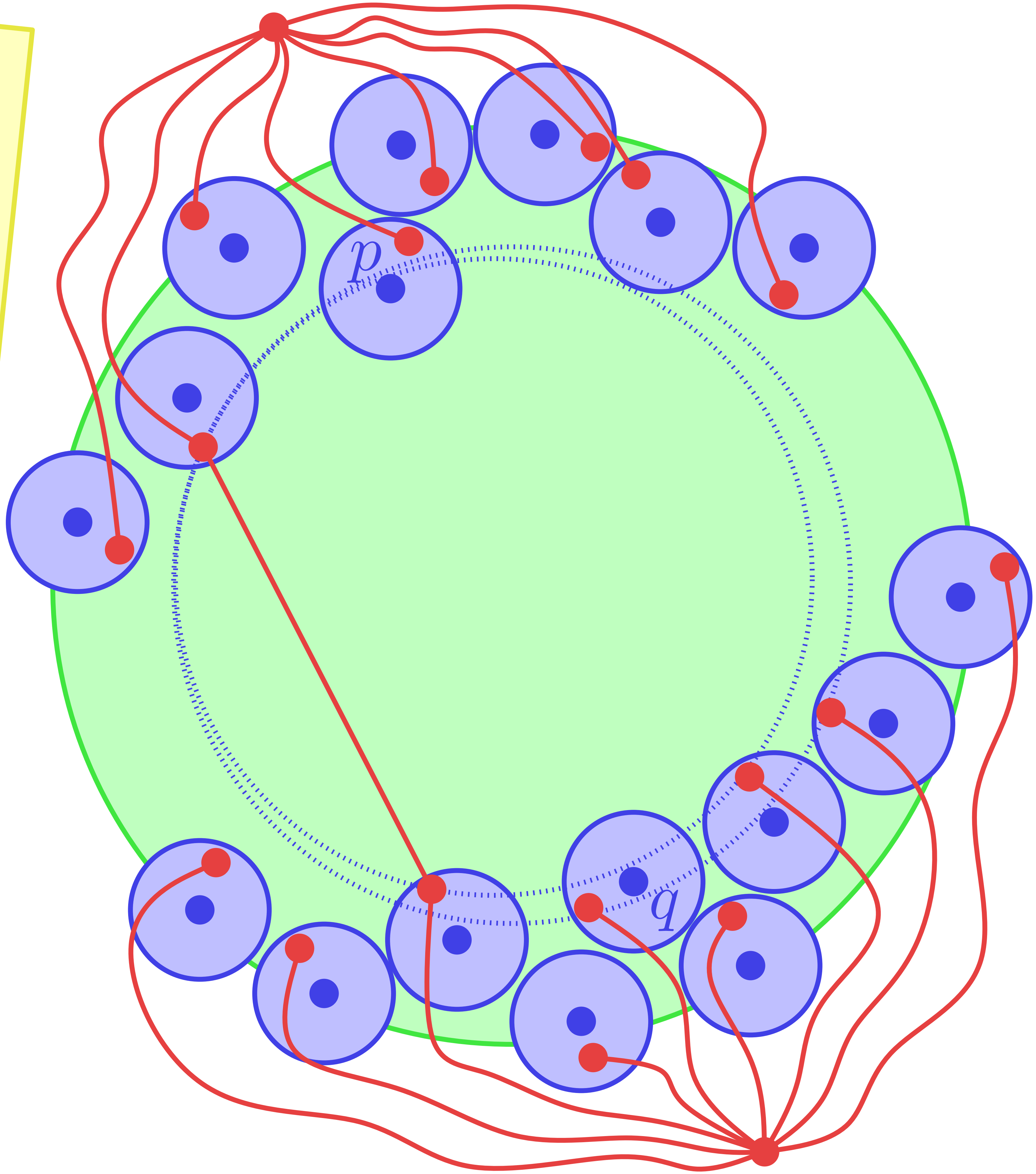
If an edge pq is *long*, all other points in C^{+}_{pq} are already connected to either p or q.



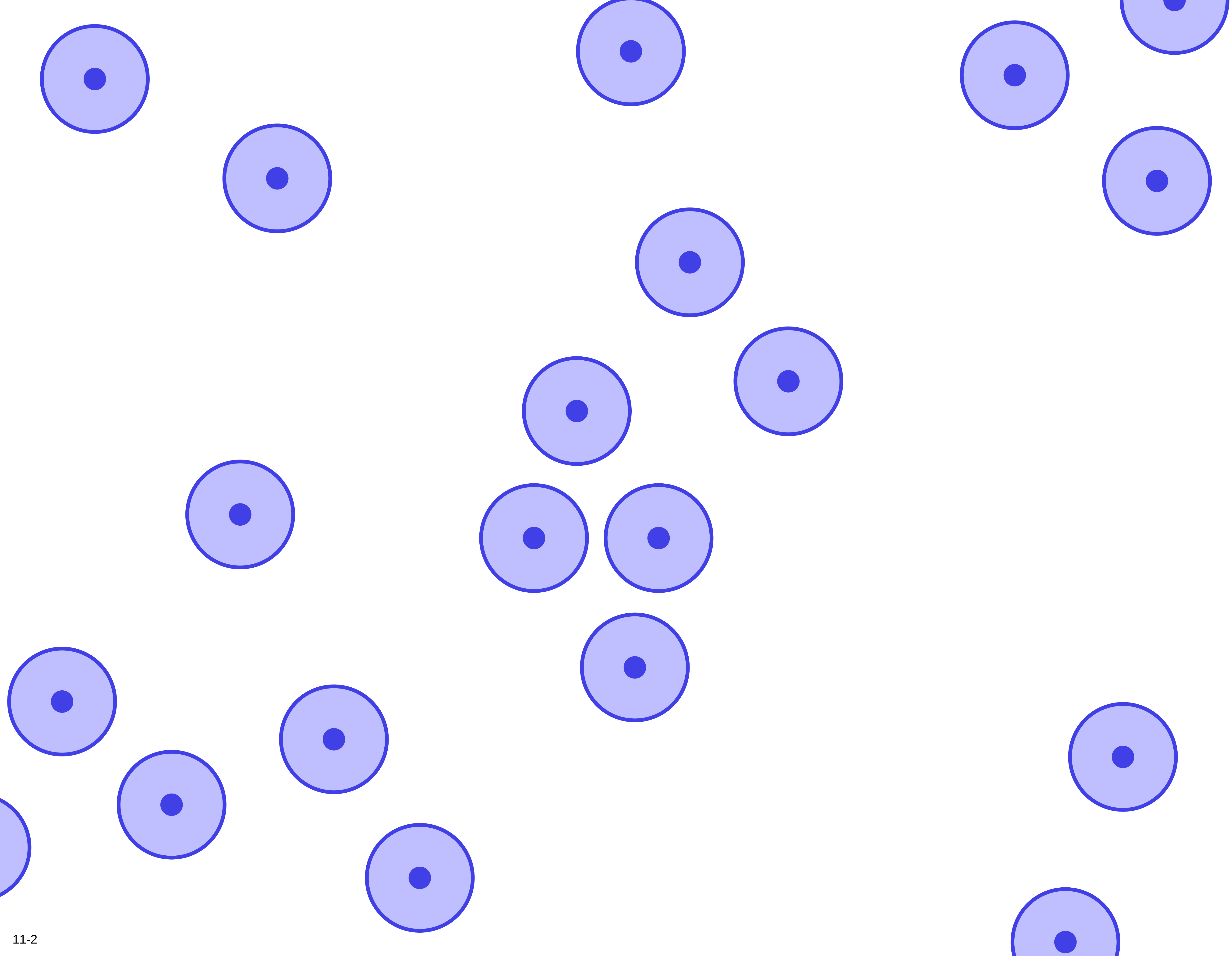
We reconstruct connectivity of the MST edges of P by length (incr

If an edge pq is *short*, the number of other points in

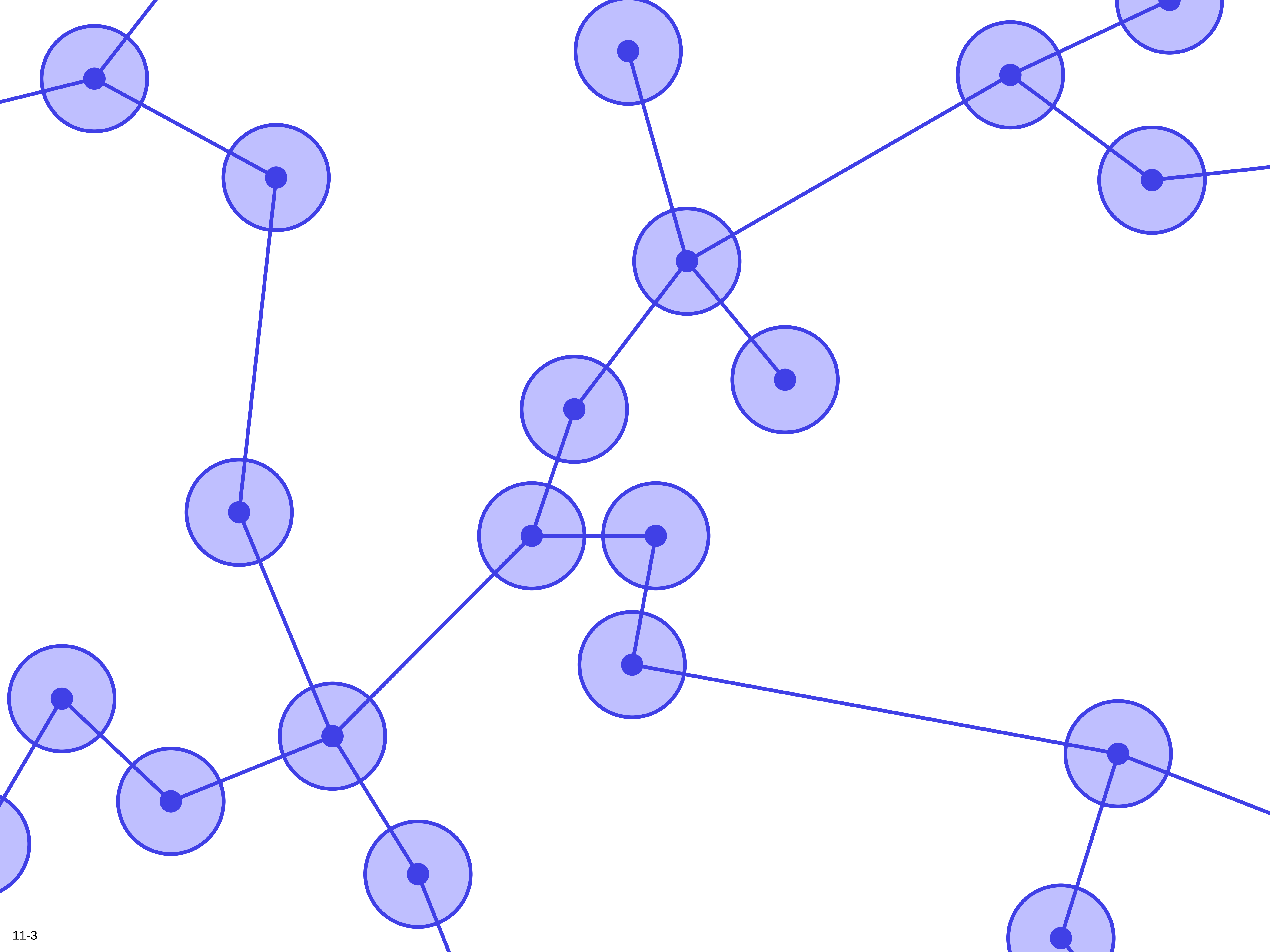
If an edge pq is *long*, all other points in C_{pq}^{+} are already connected to either p or q.



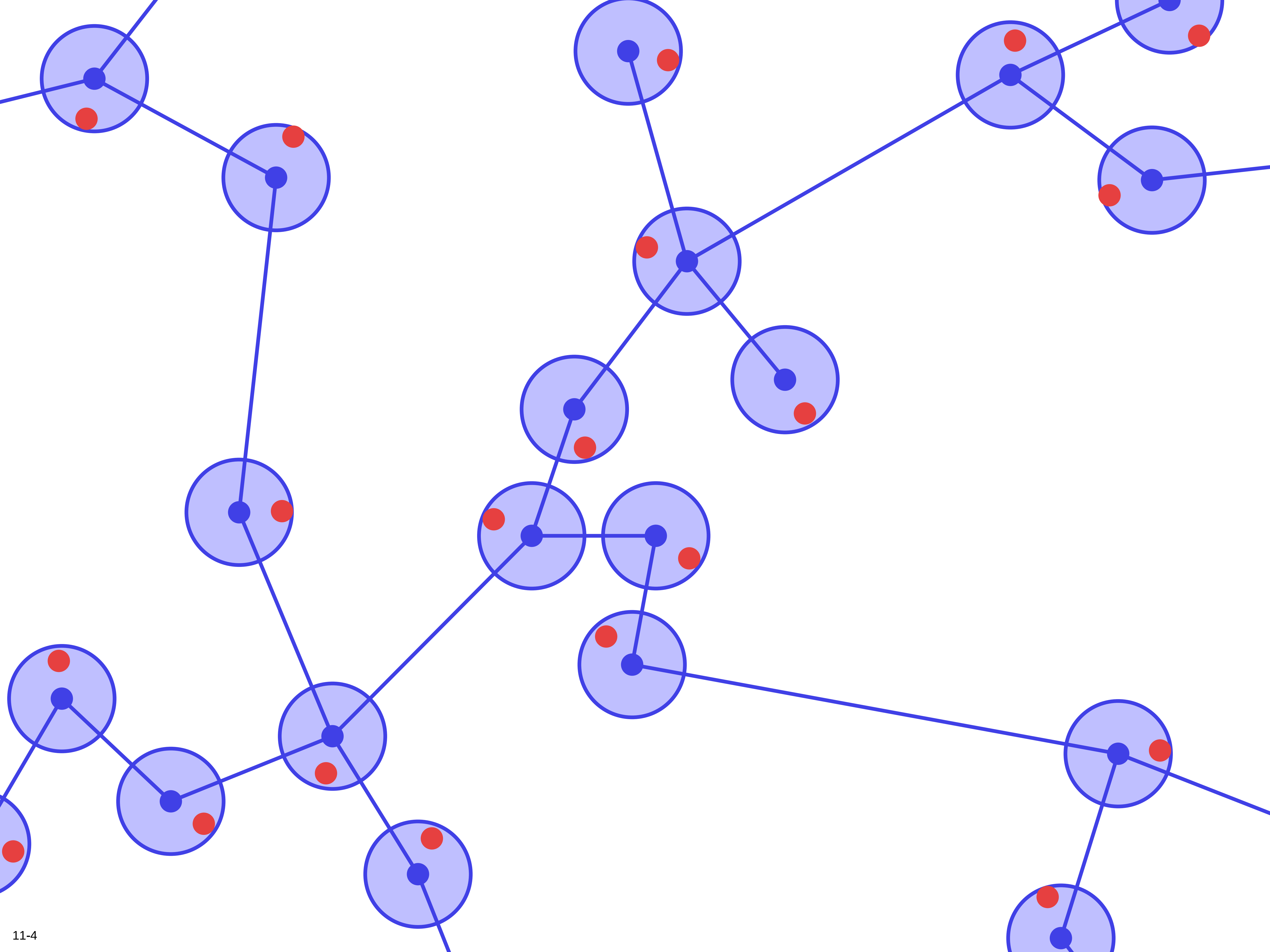
Let's put it together and reconstruct the Delaunay triangulation!

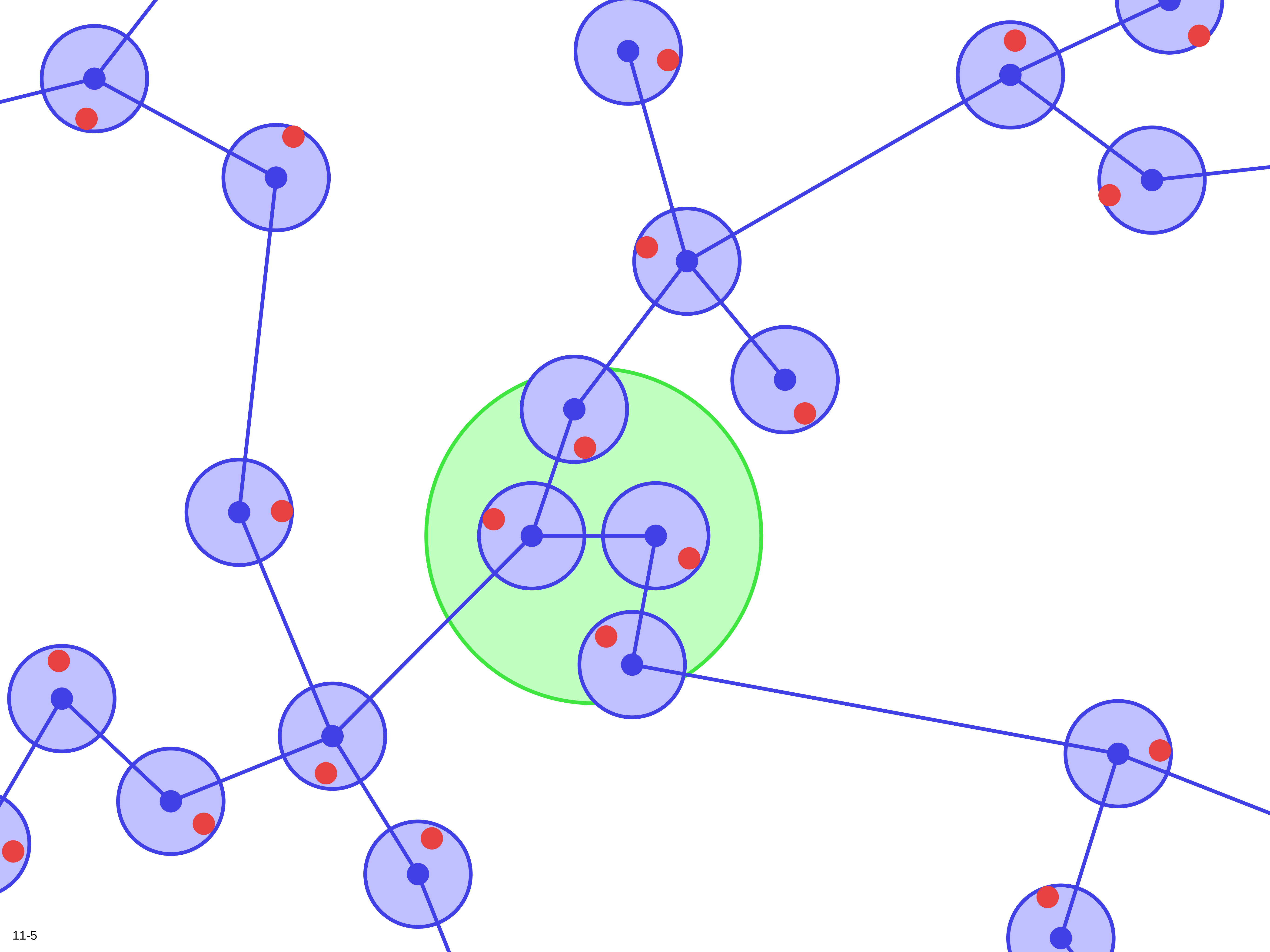


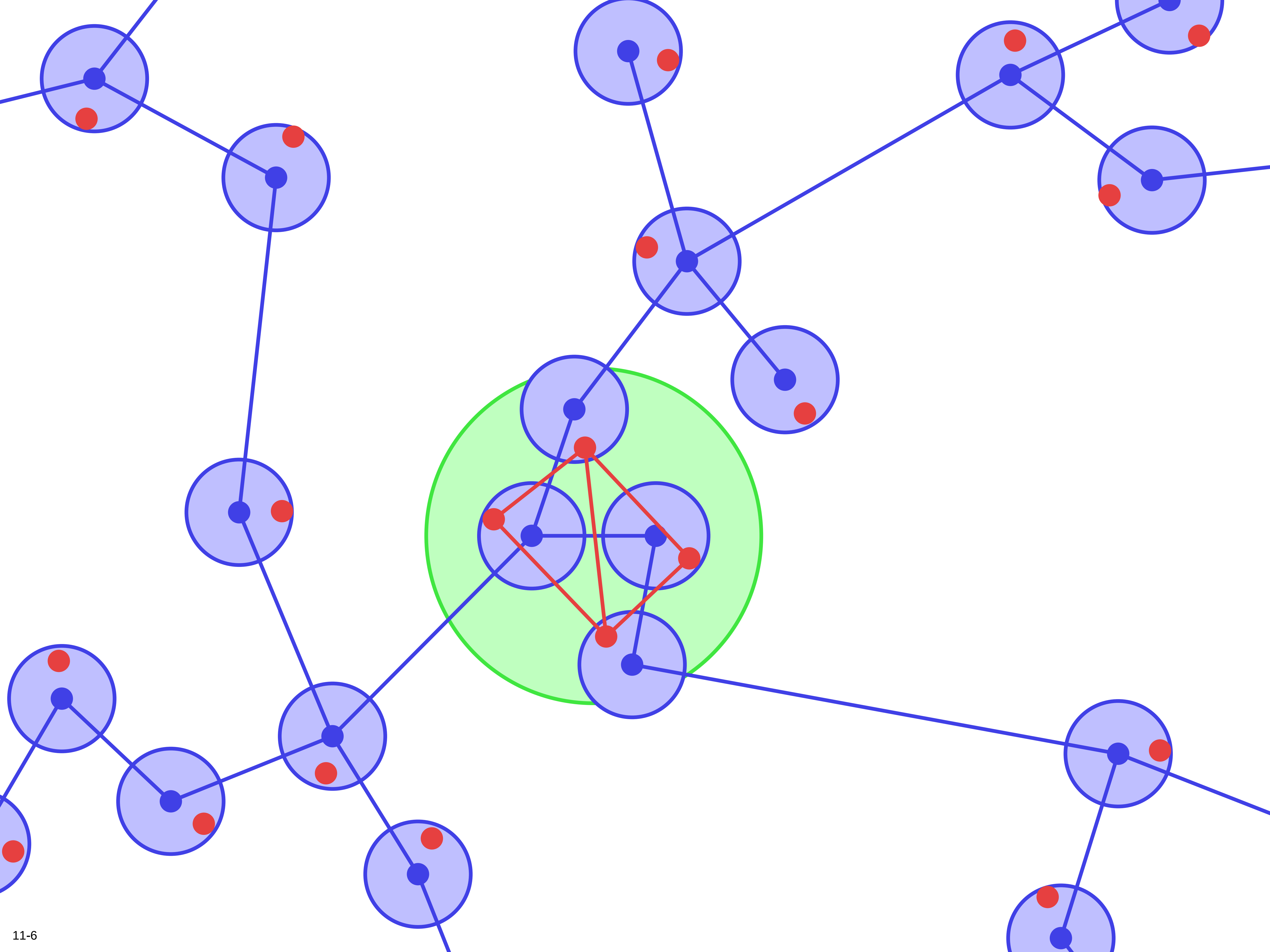
11-2

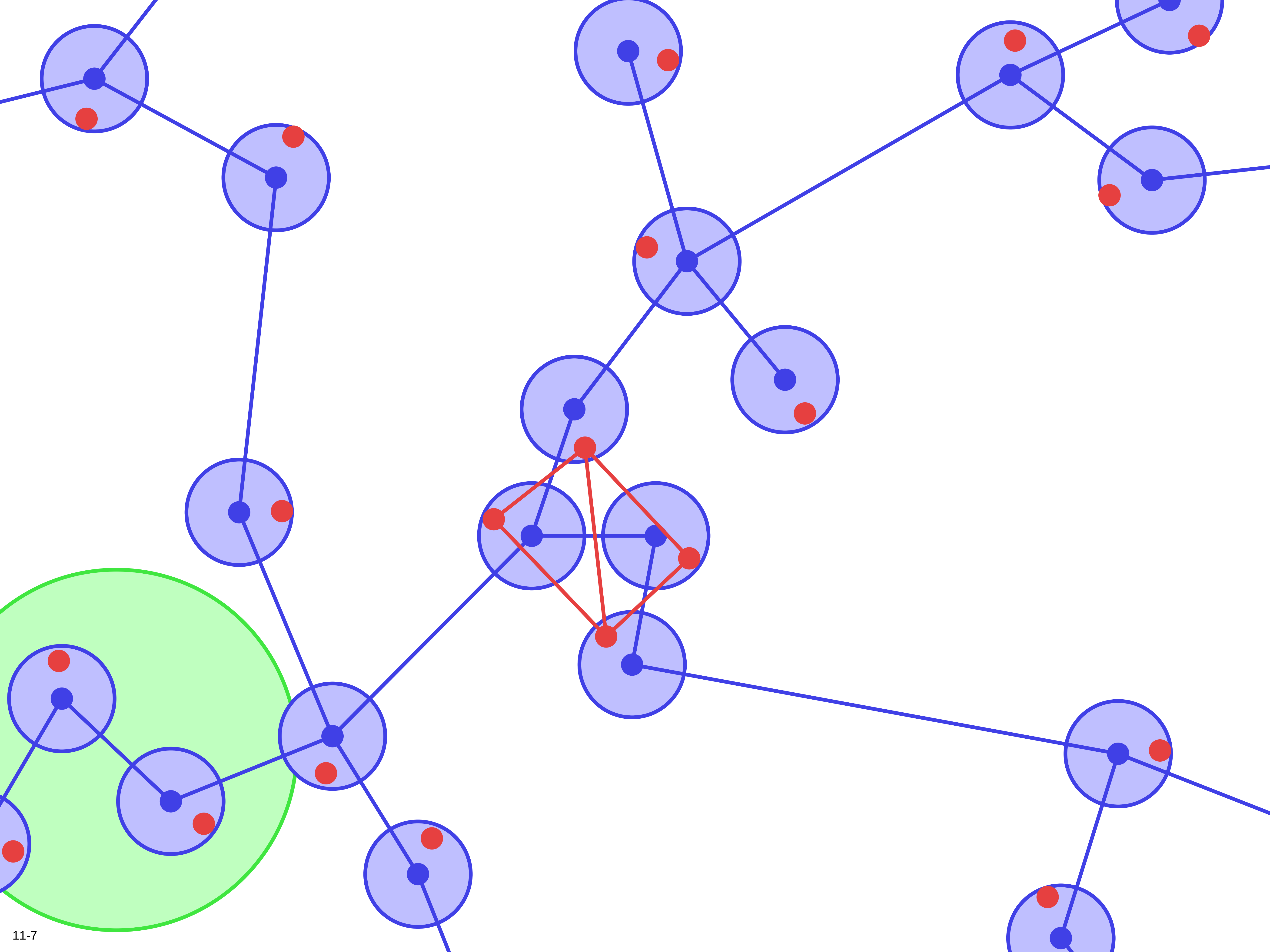


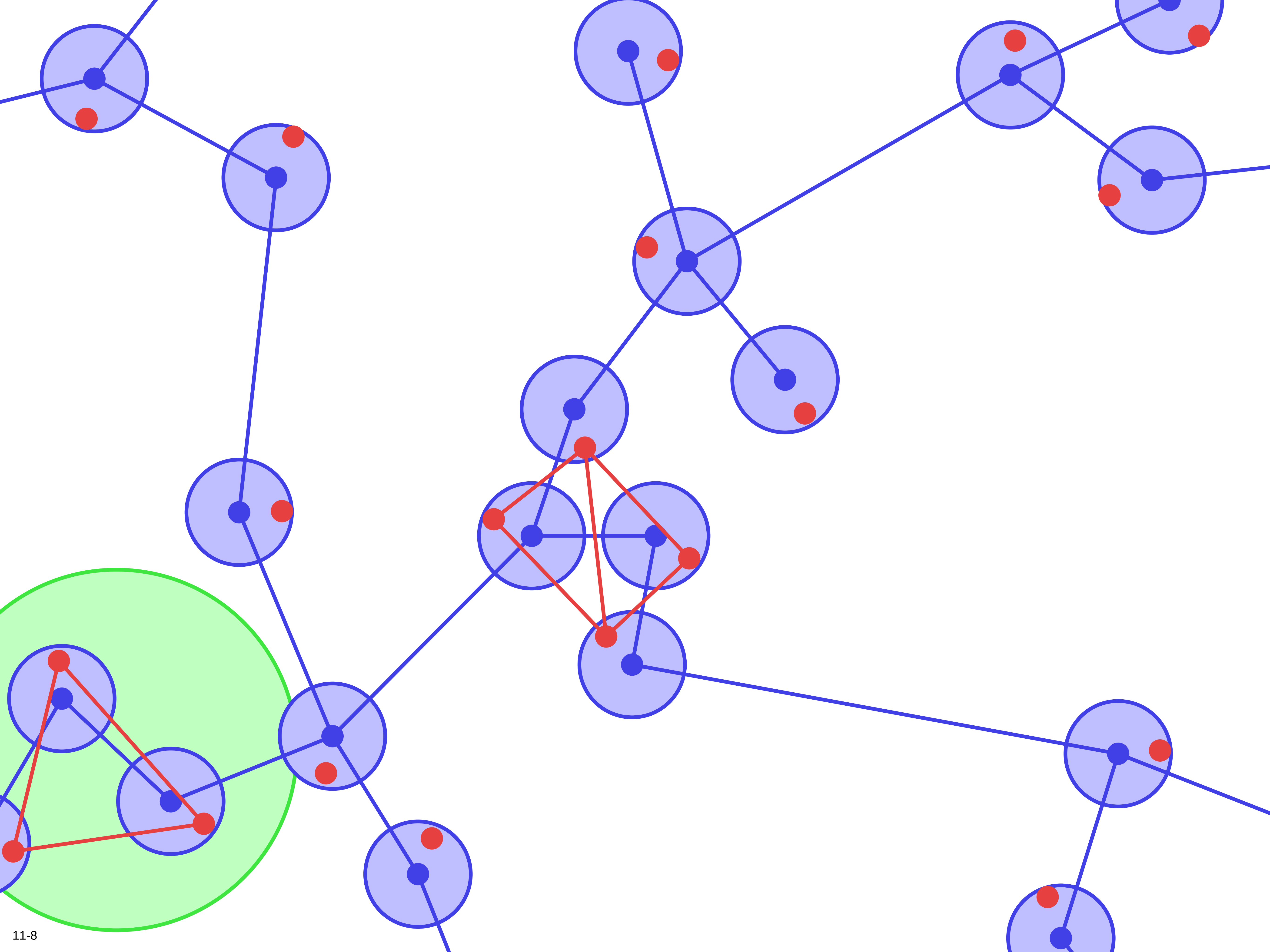
11-3



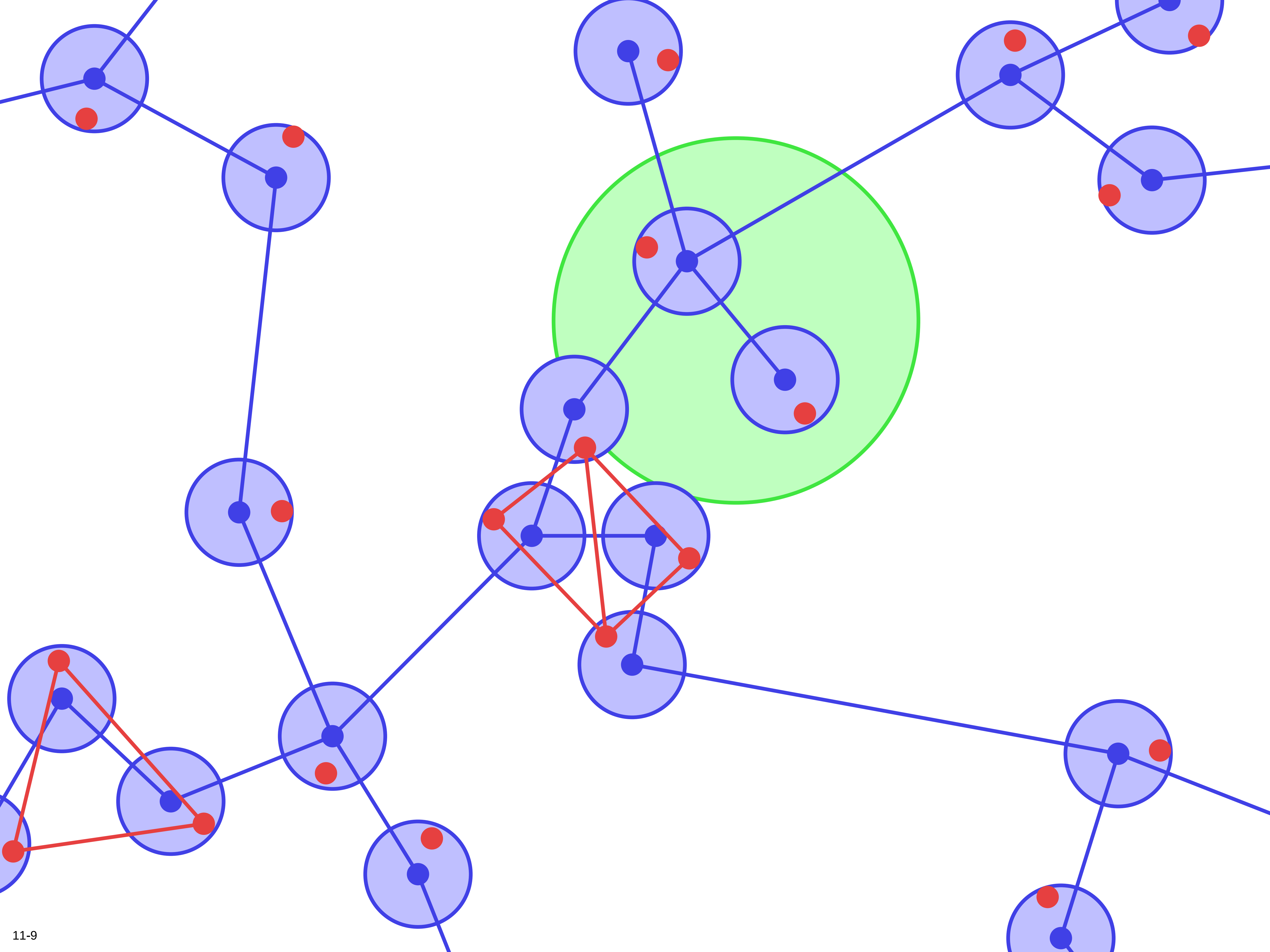


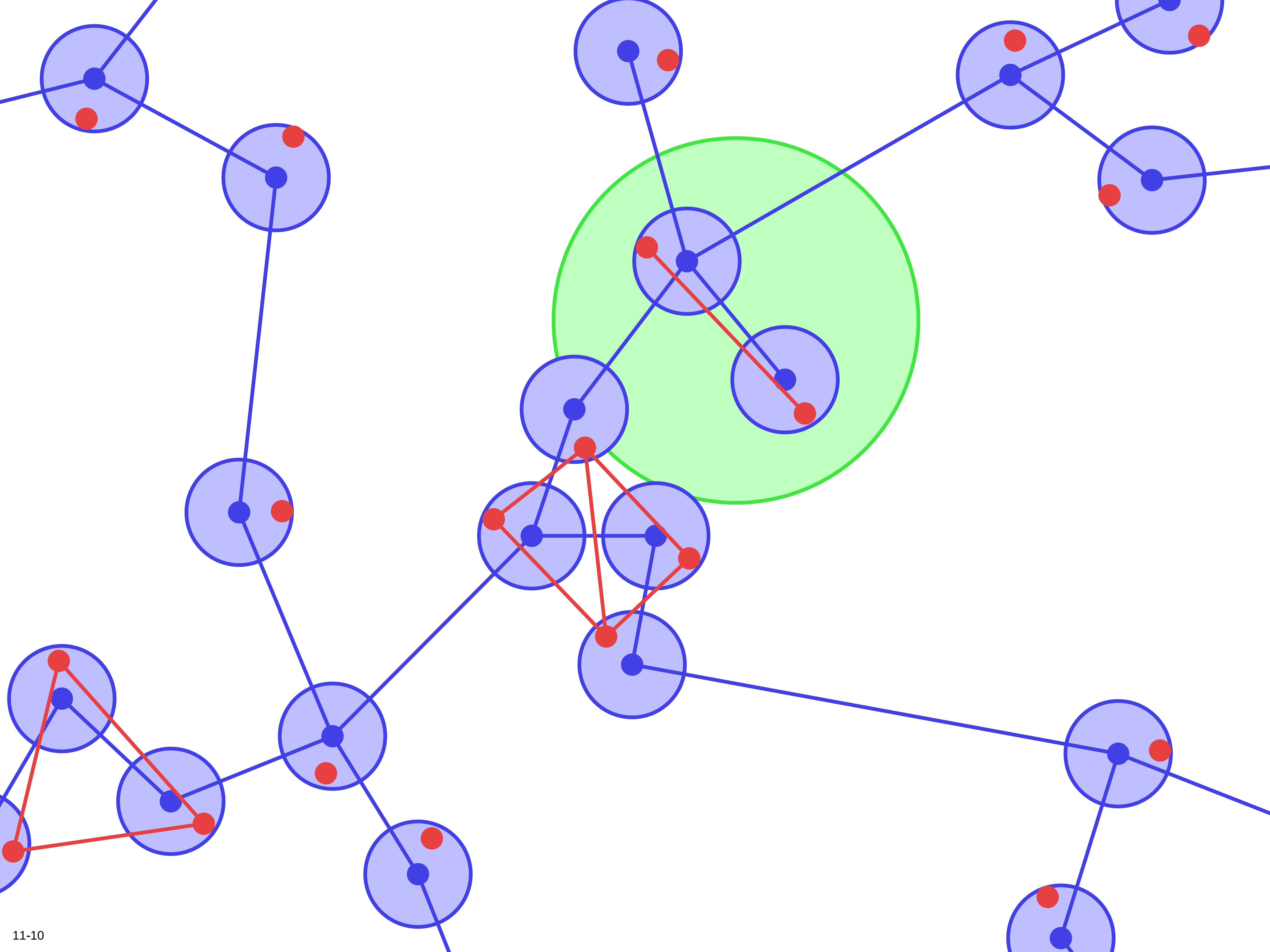


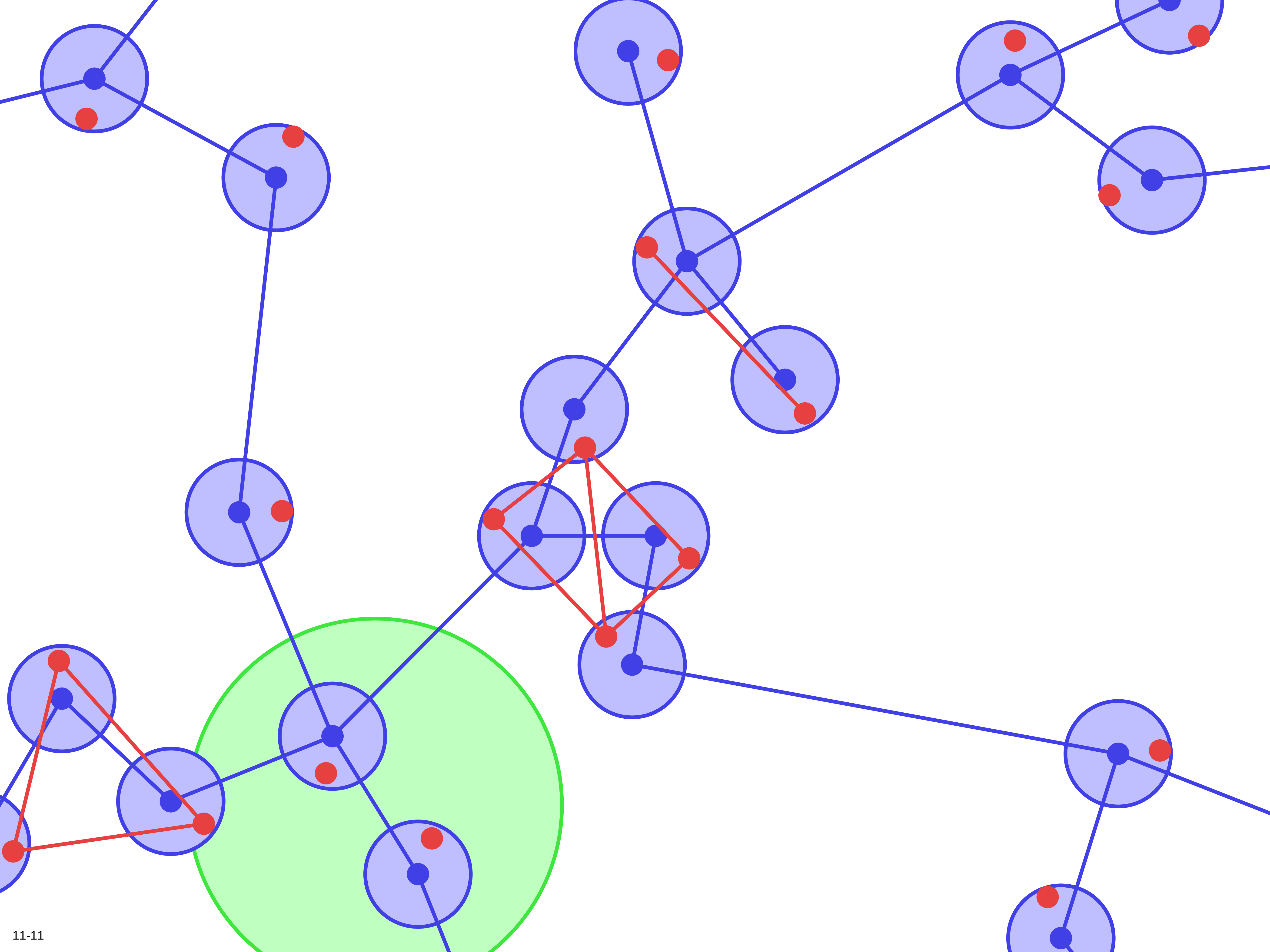

11-7

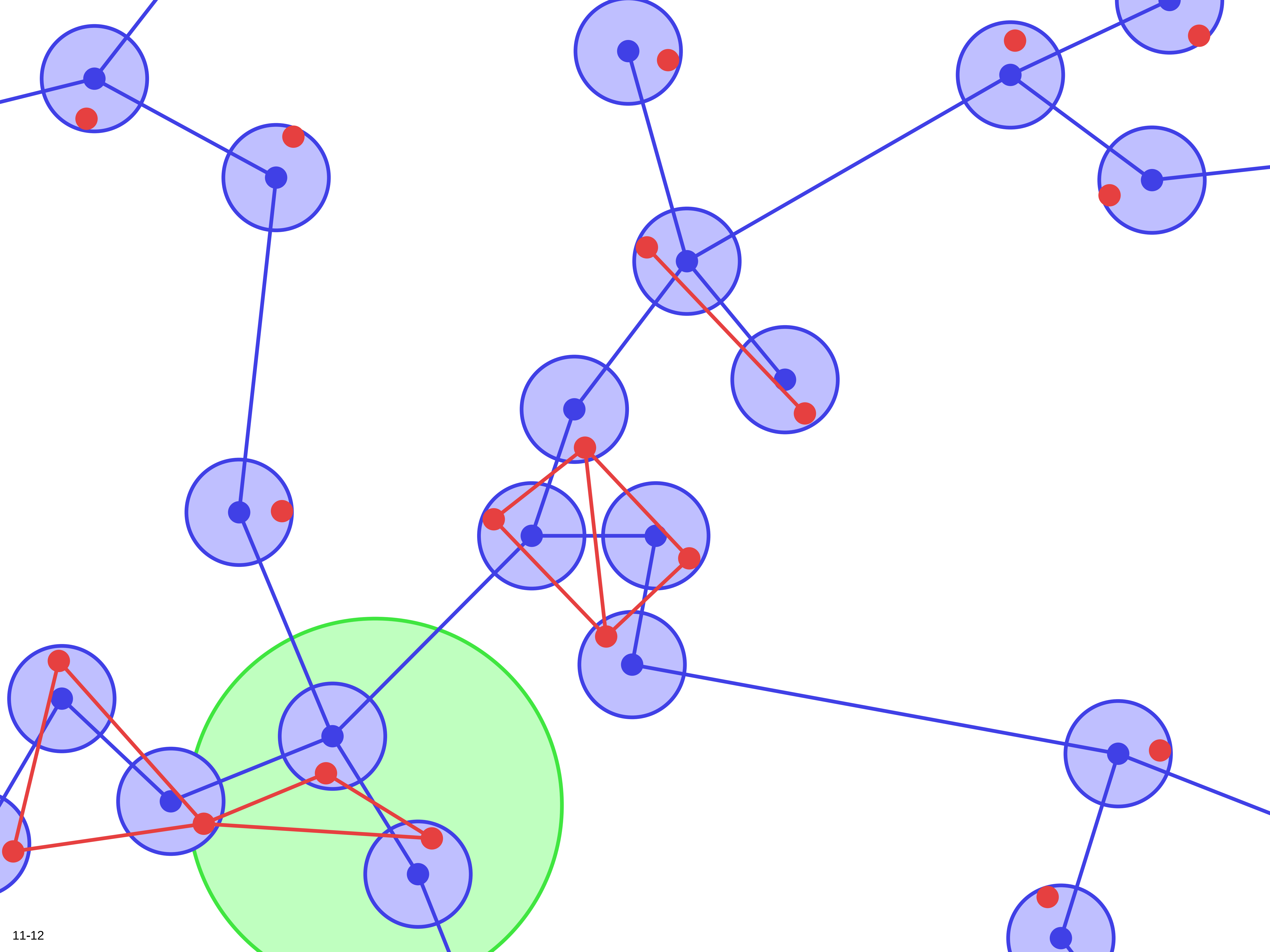


11-8

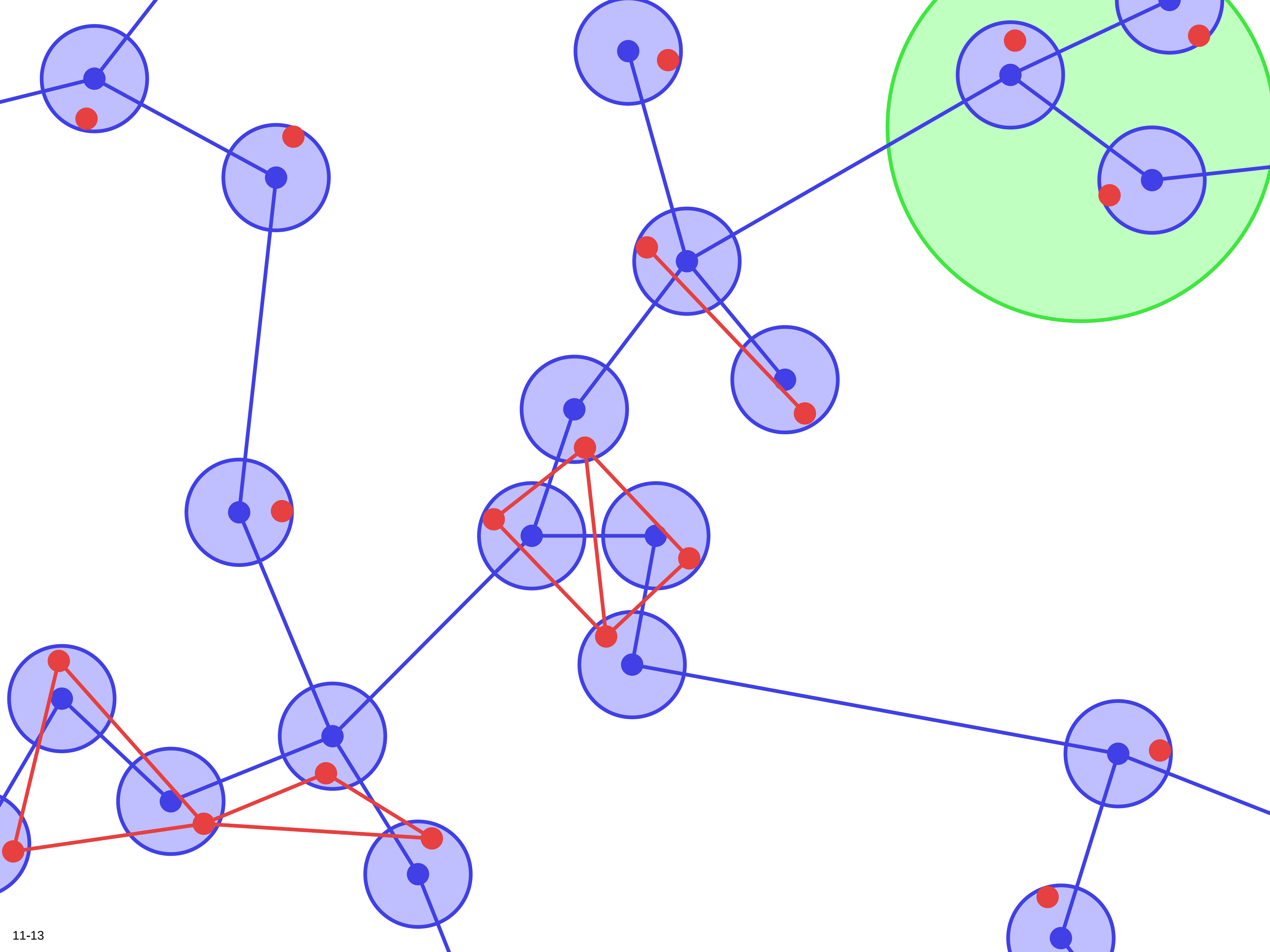


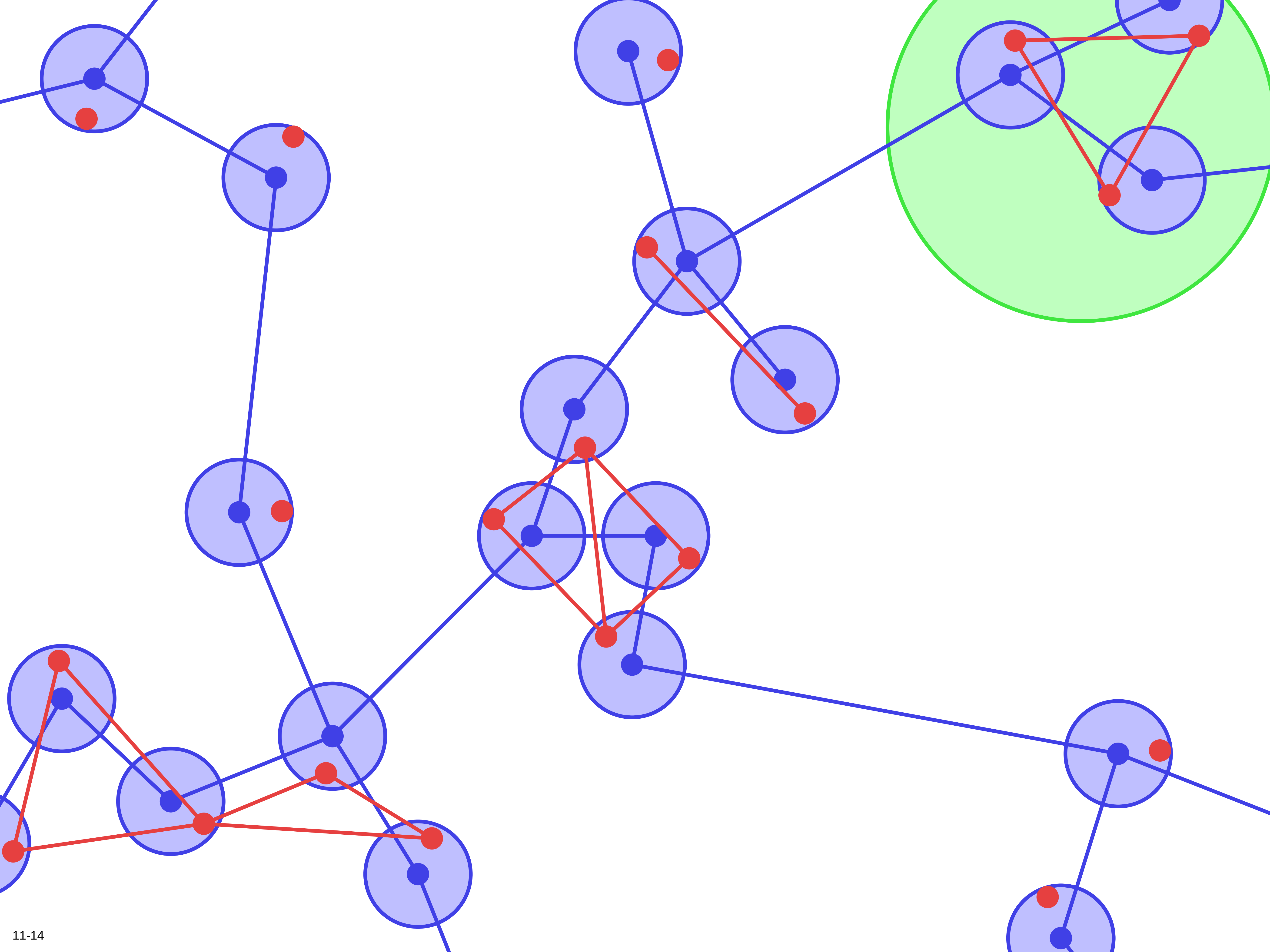




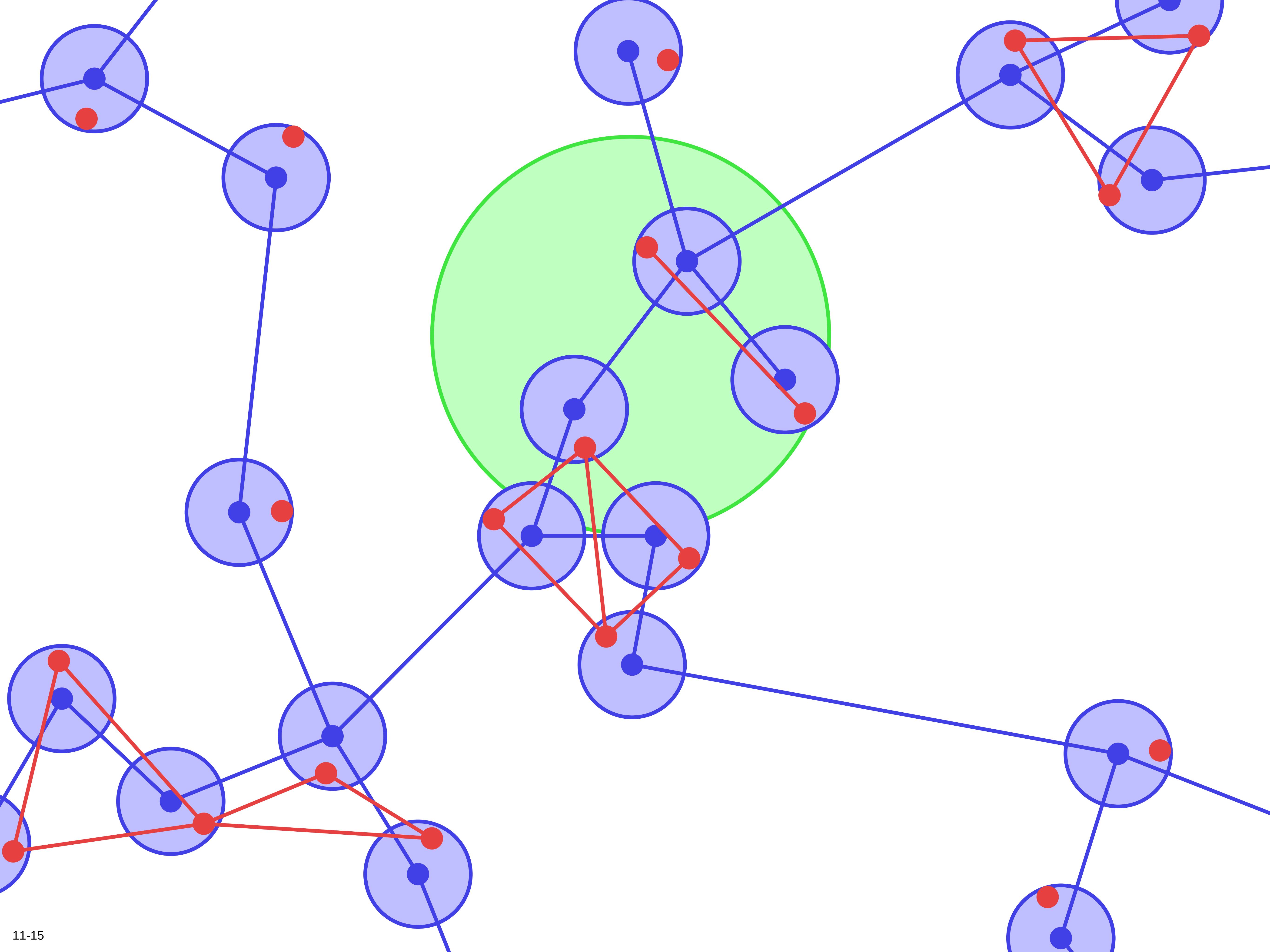


11-12

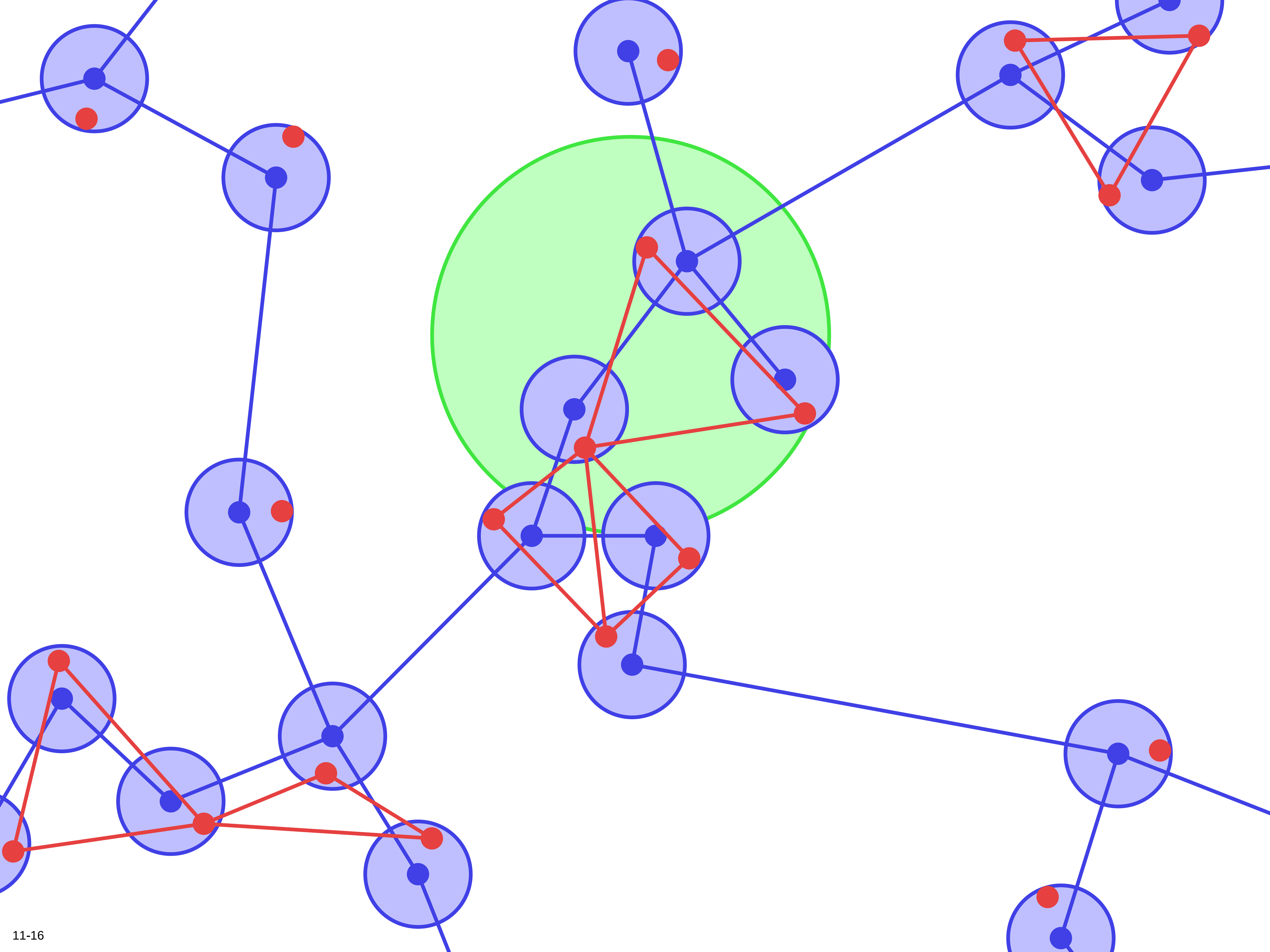


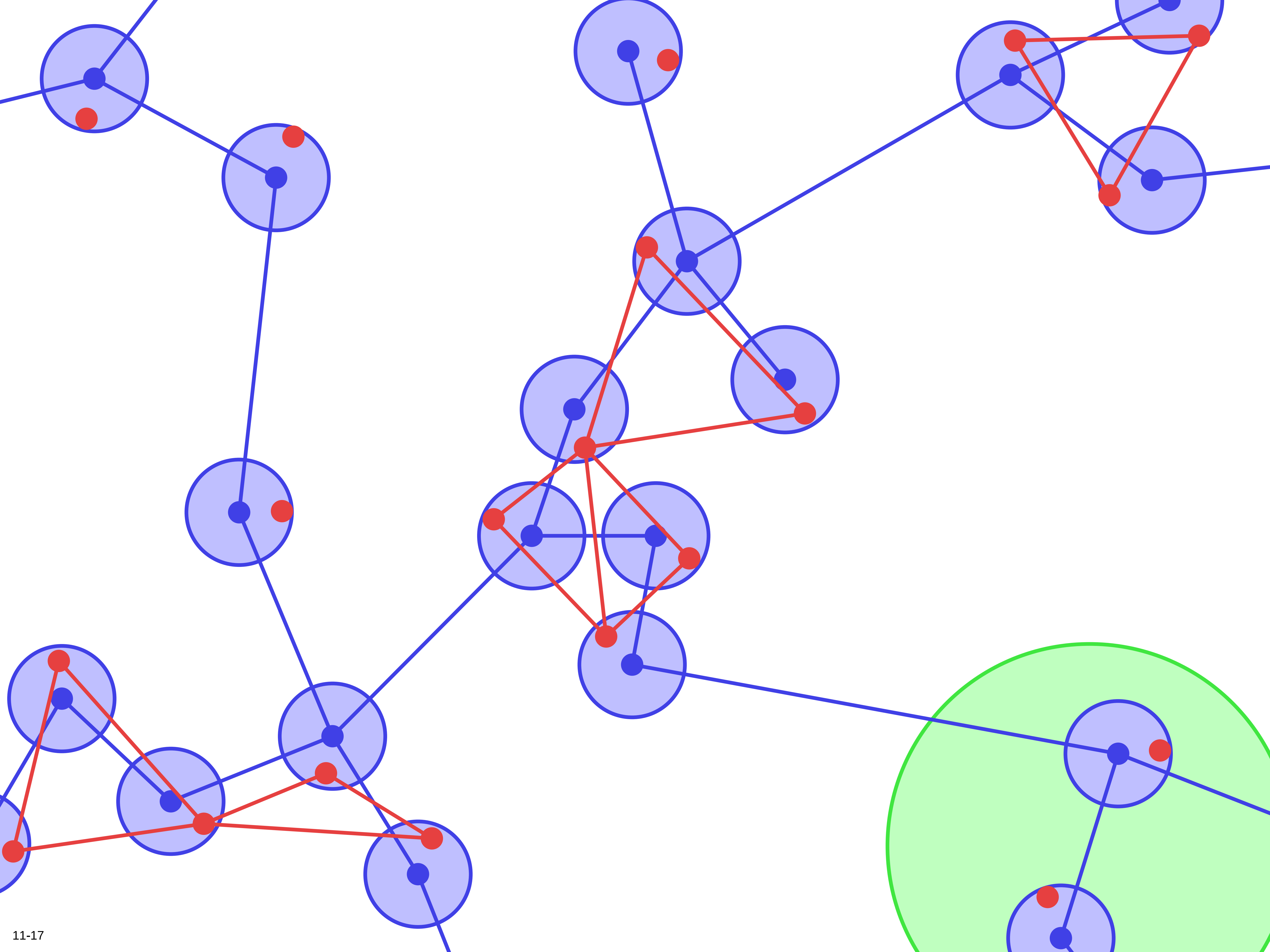


11-14

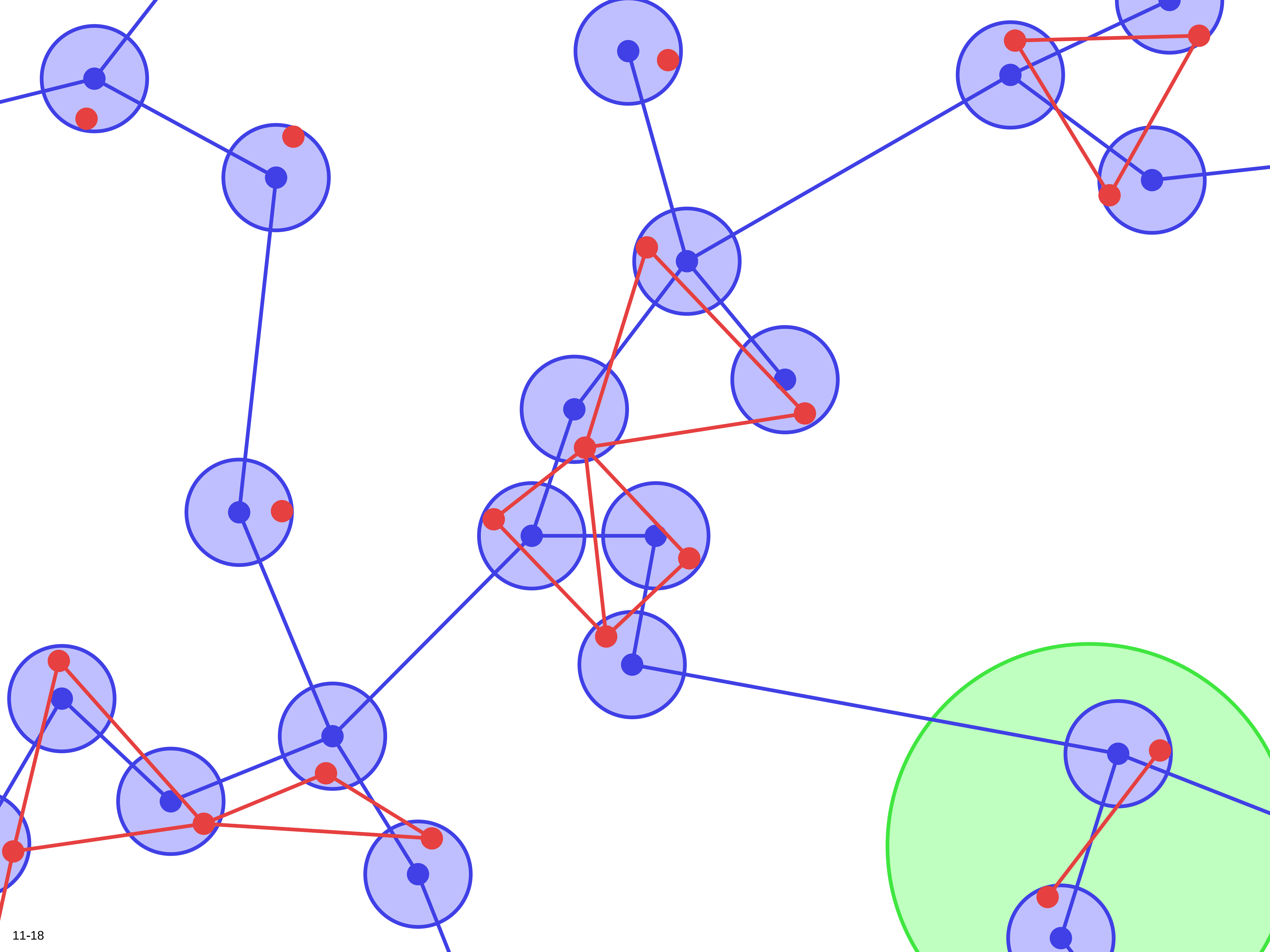


11-15

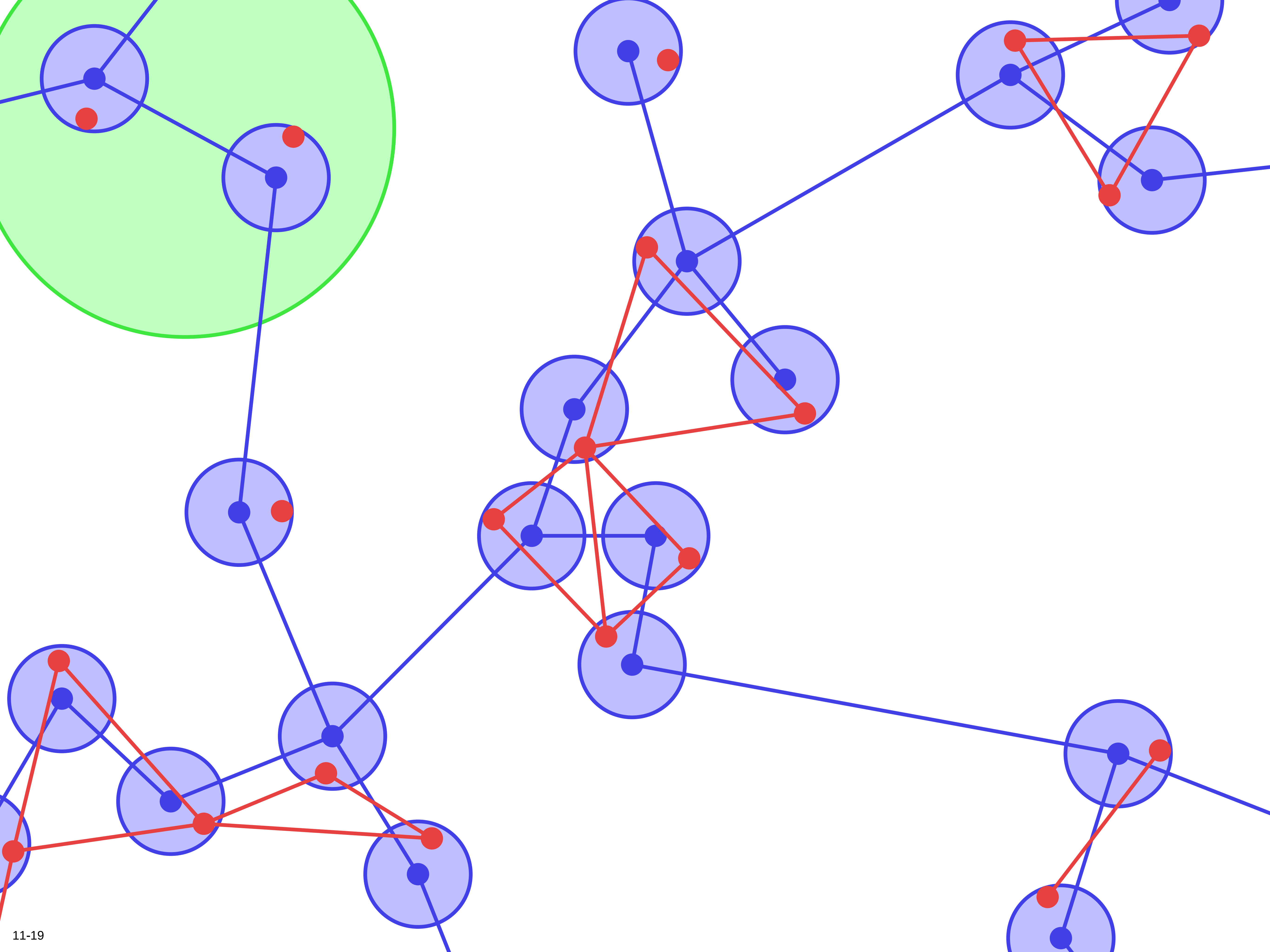


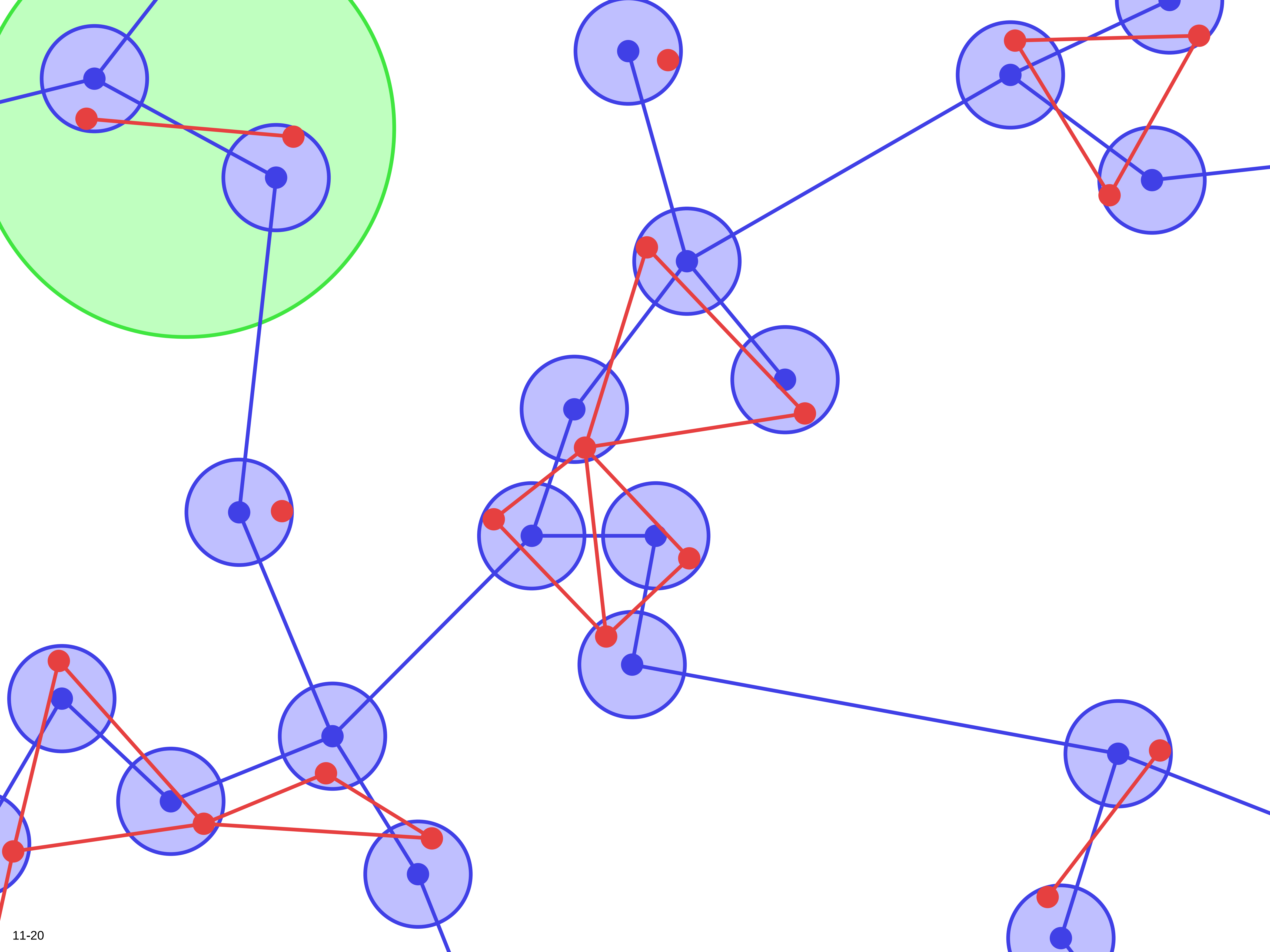


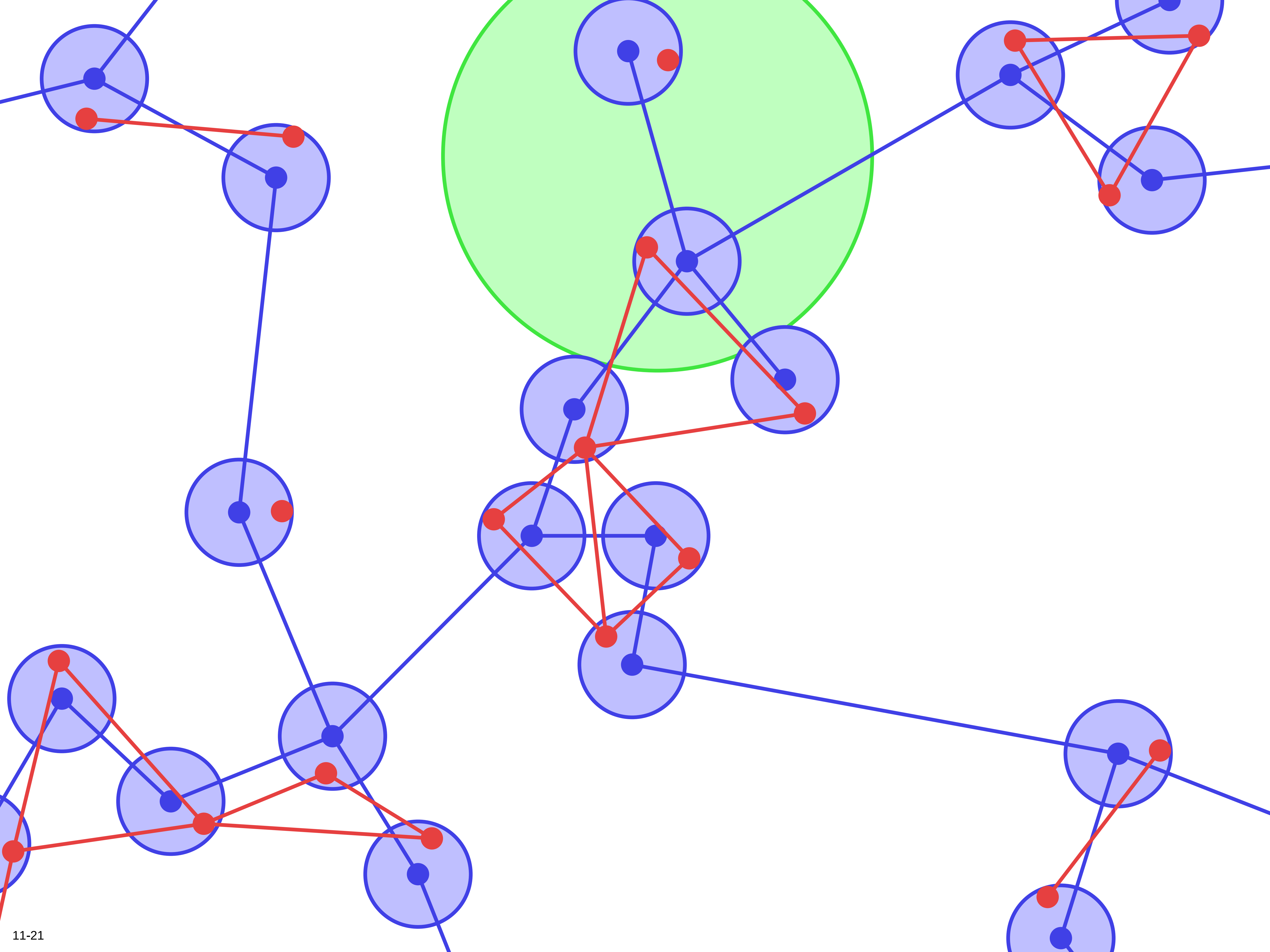
11-17



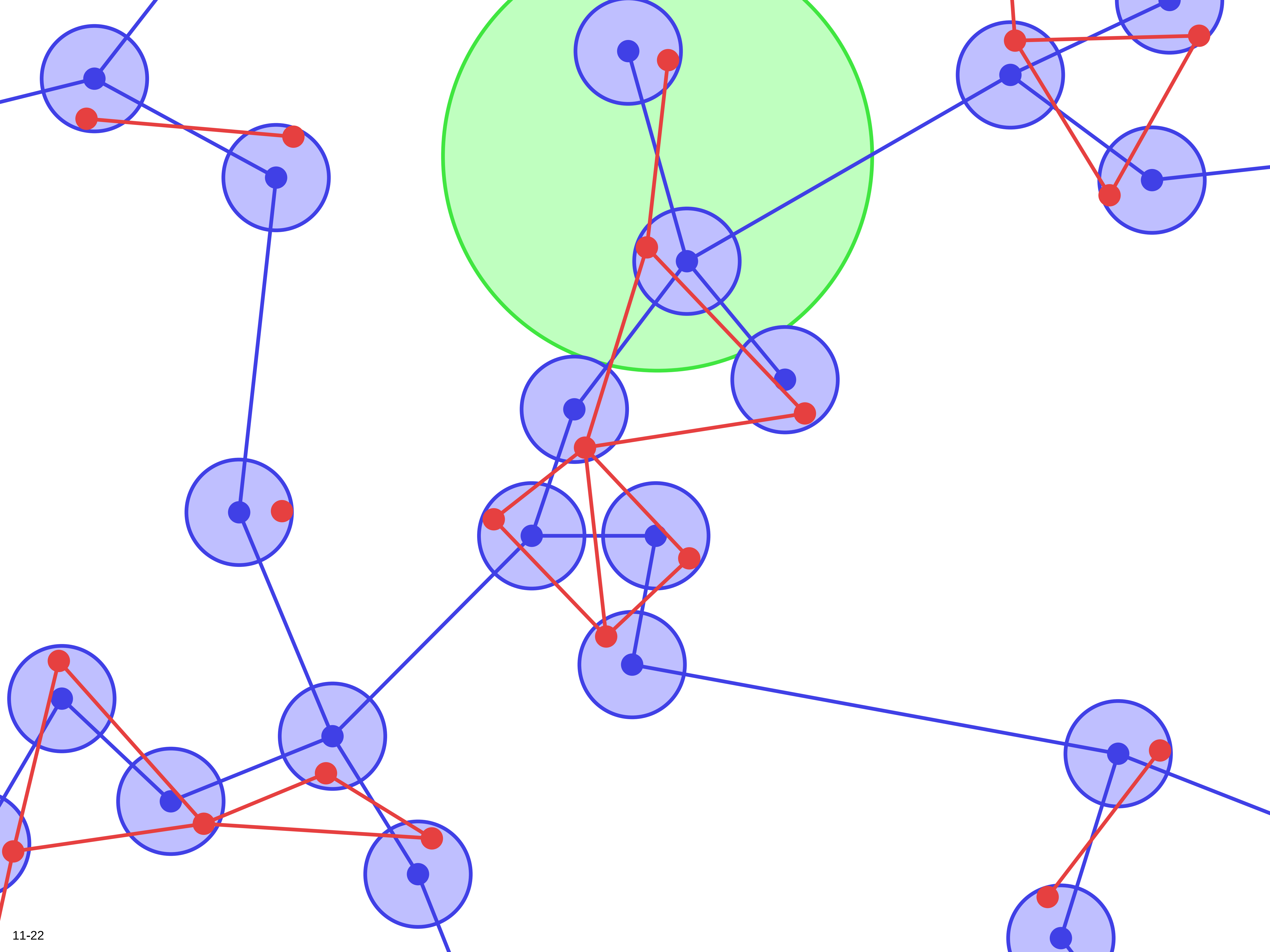
11-18



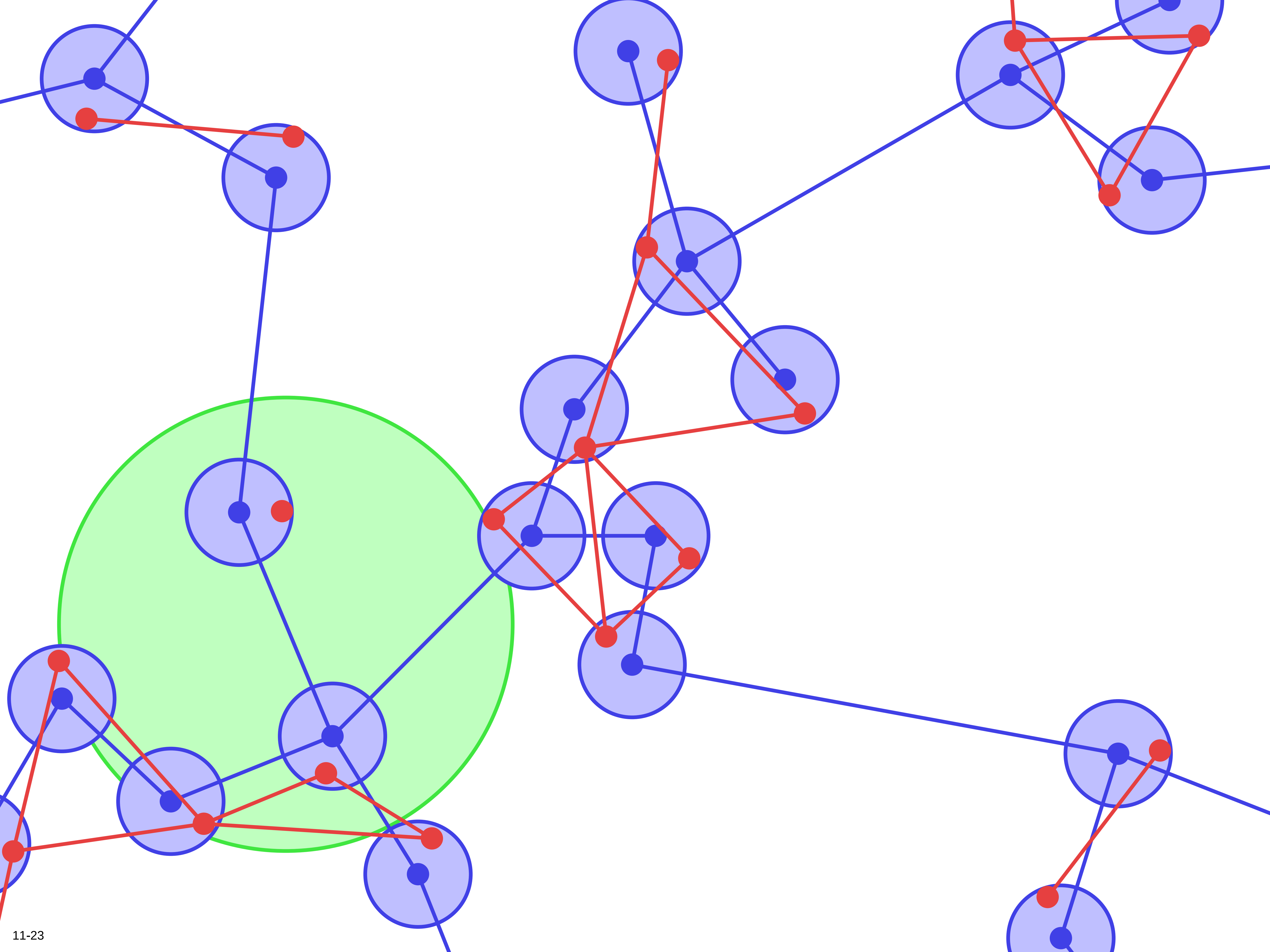


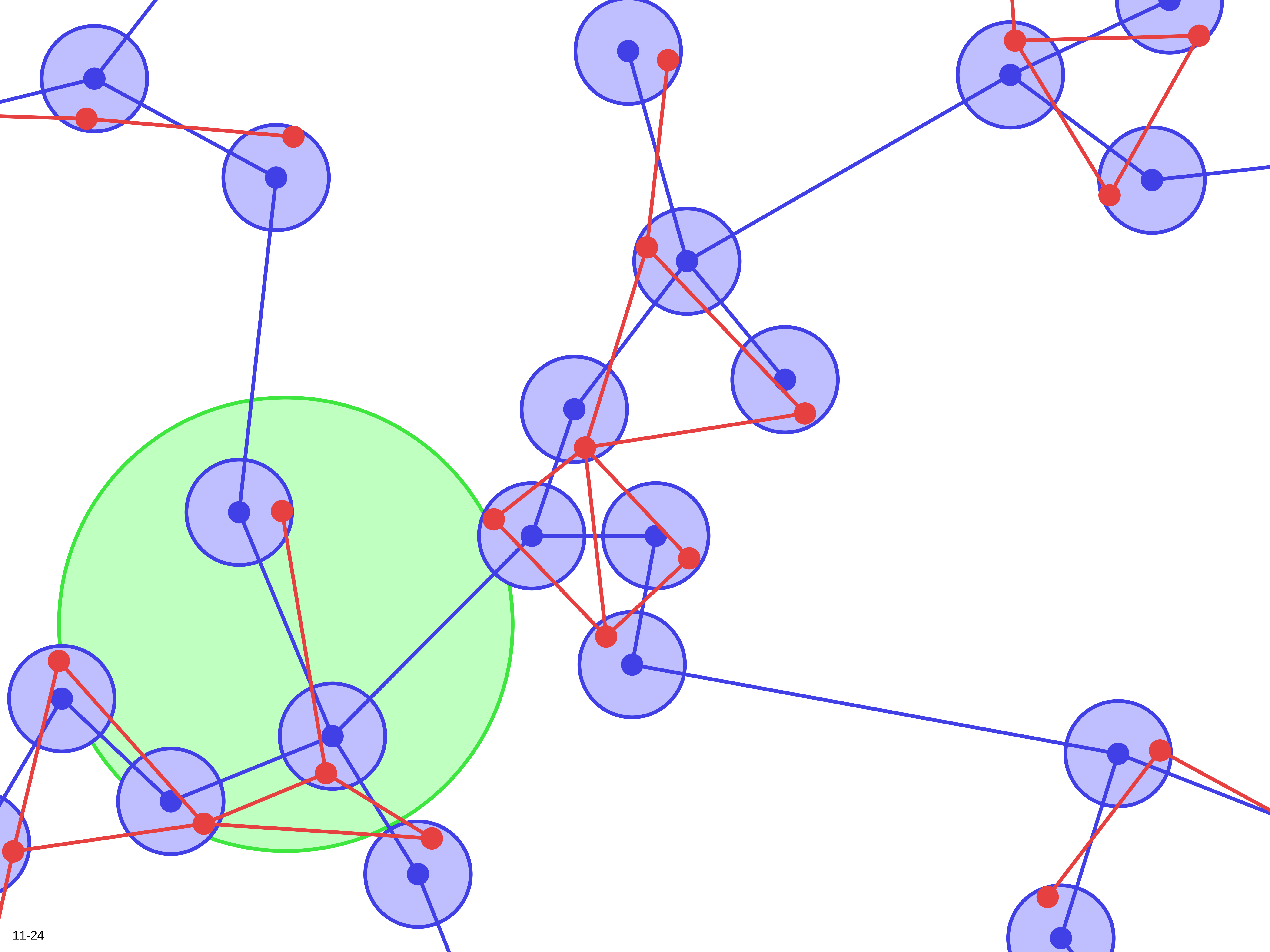


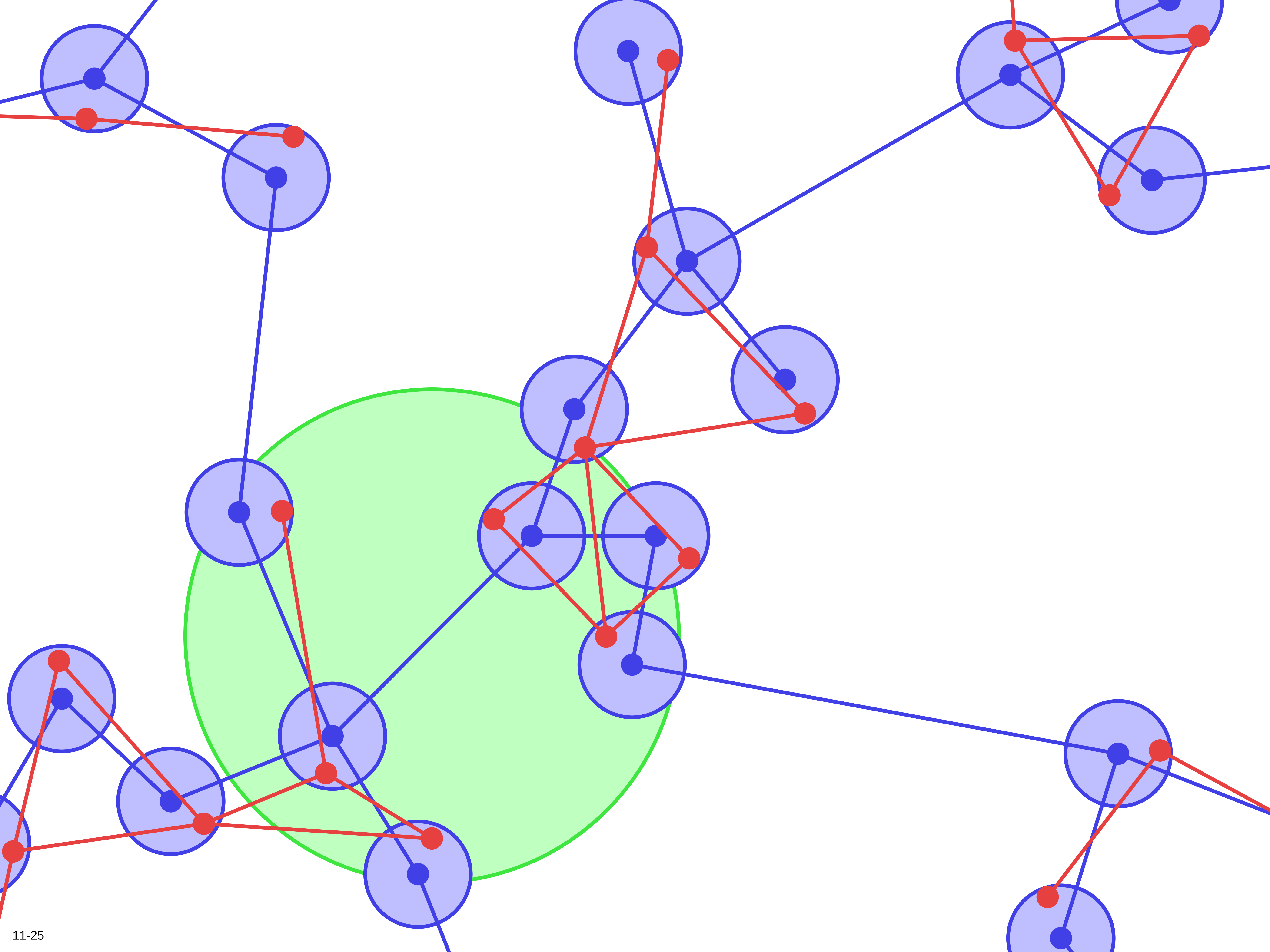
11-21



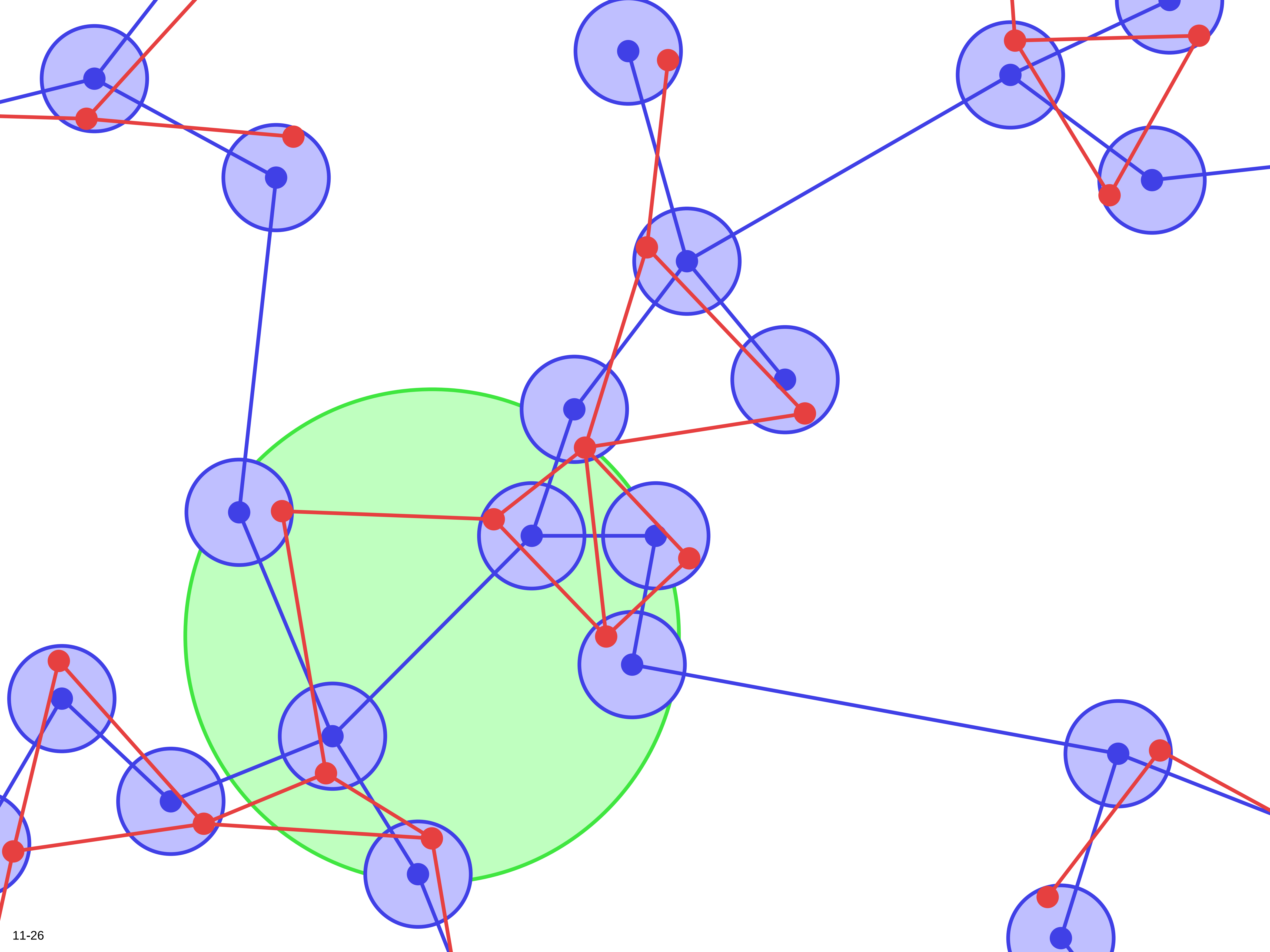
11-22



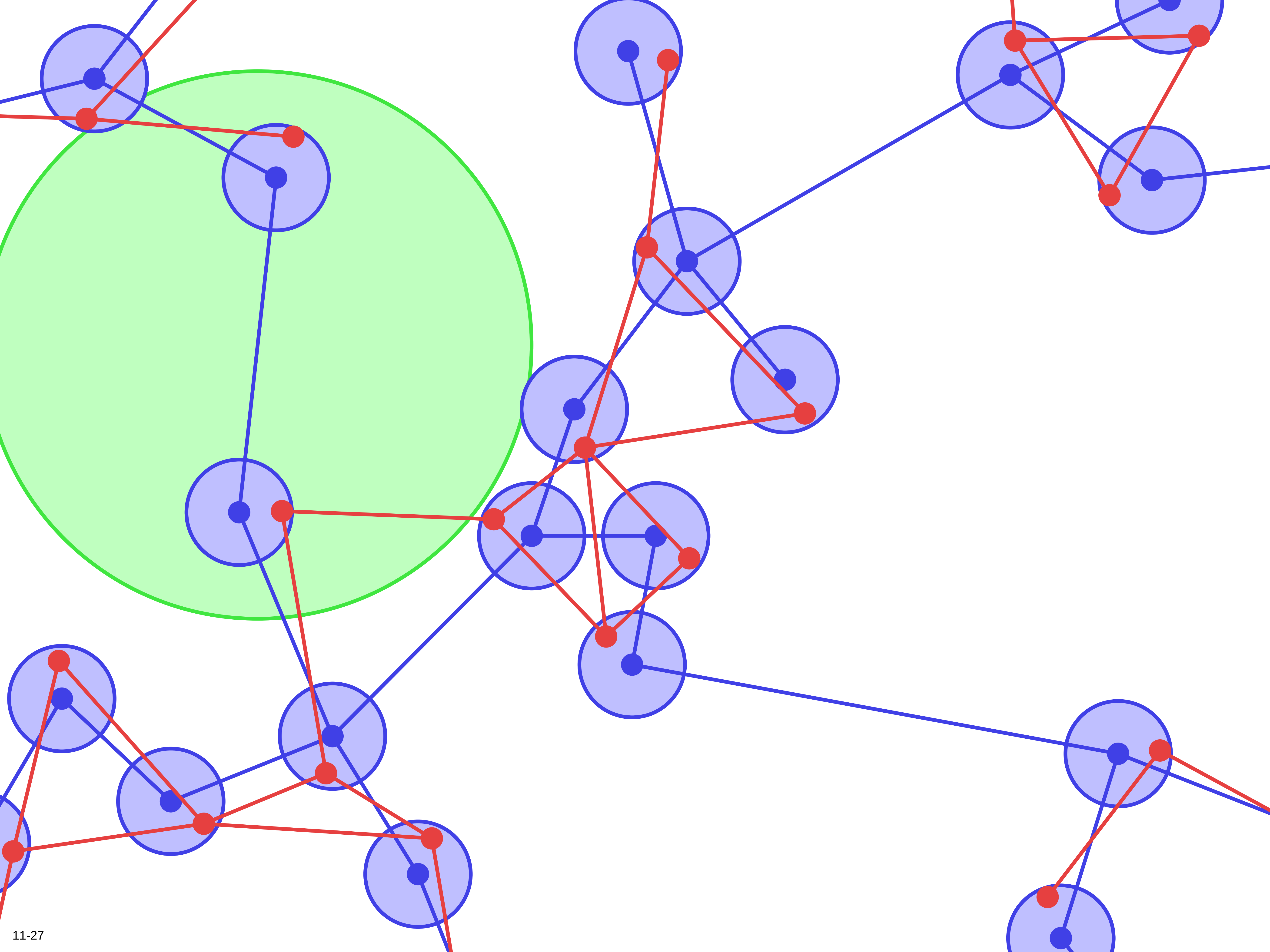


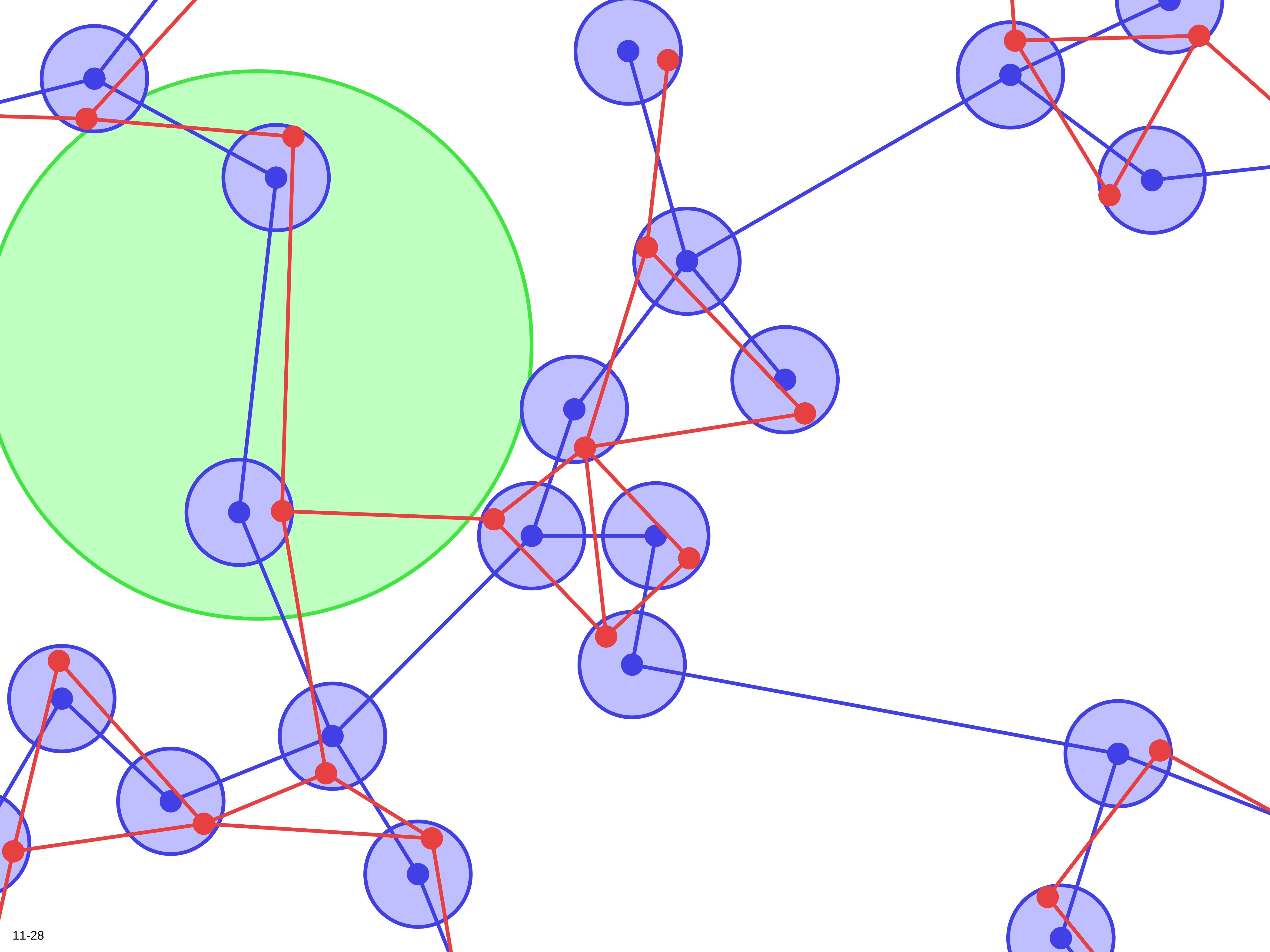


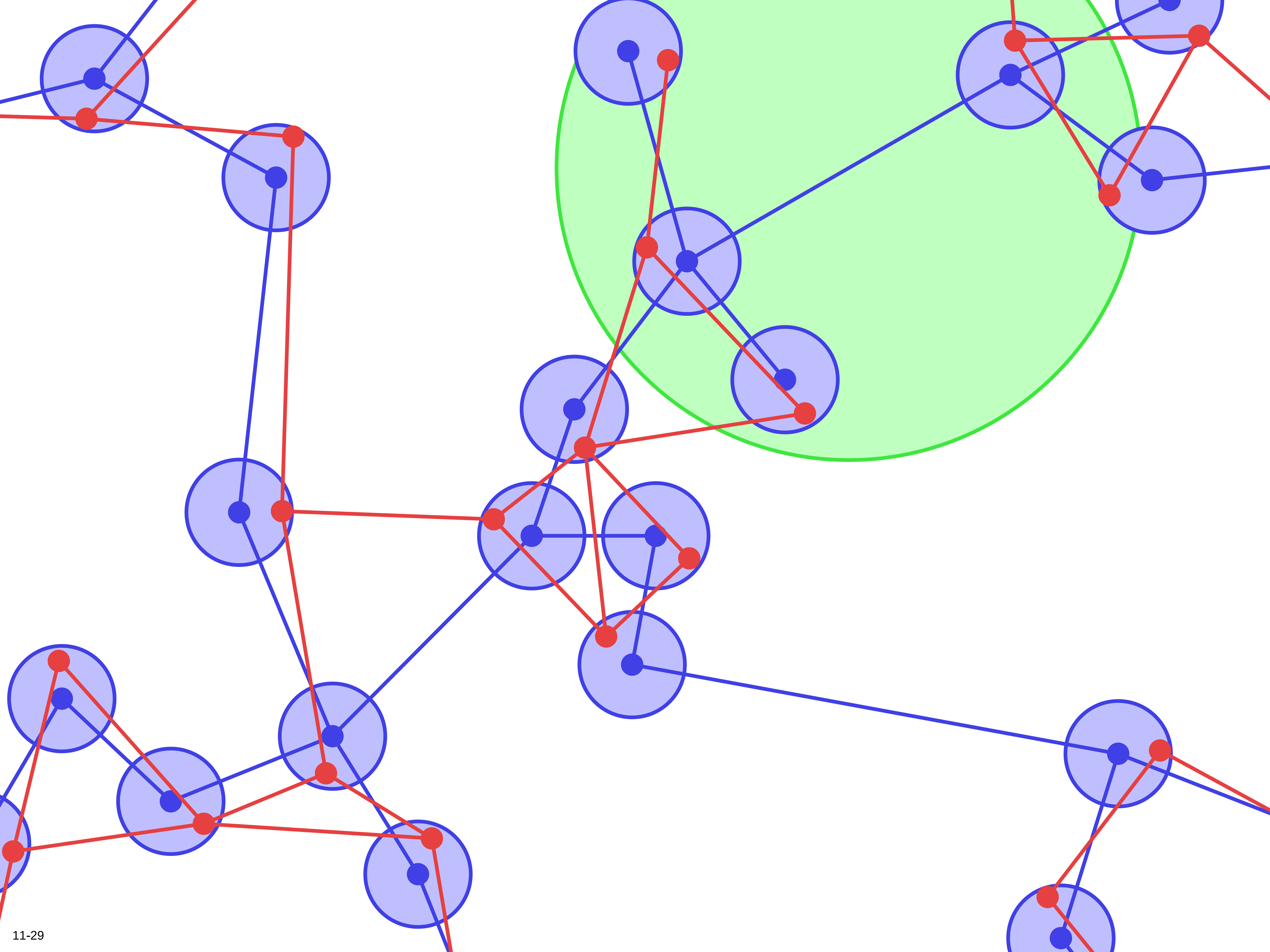
11-25



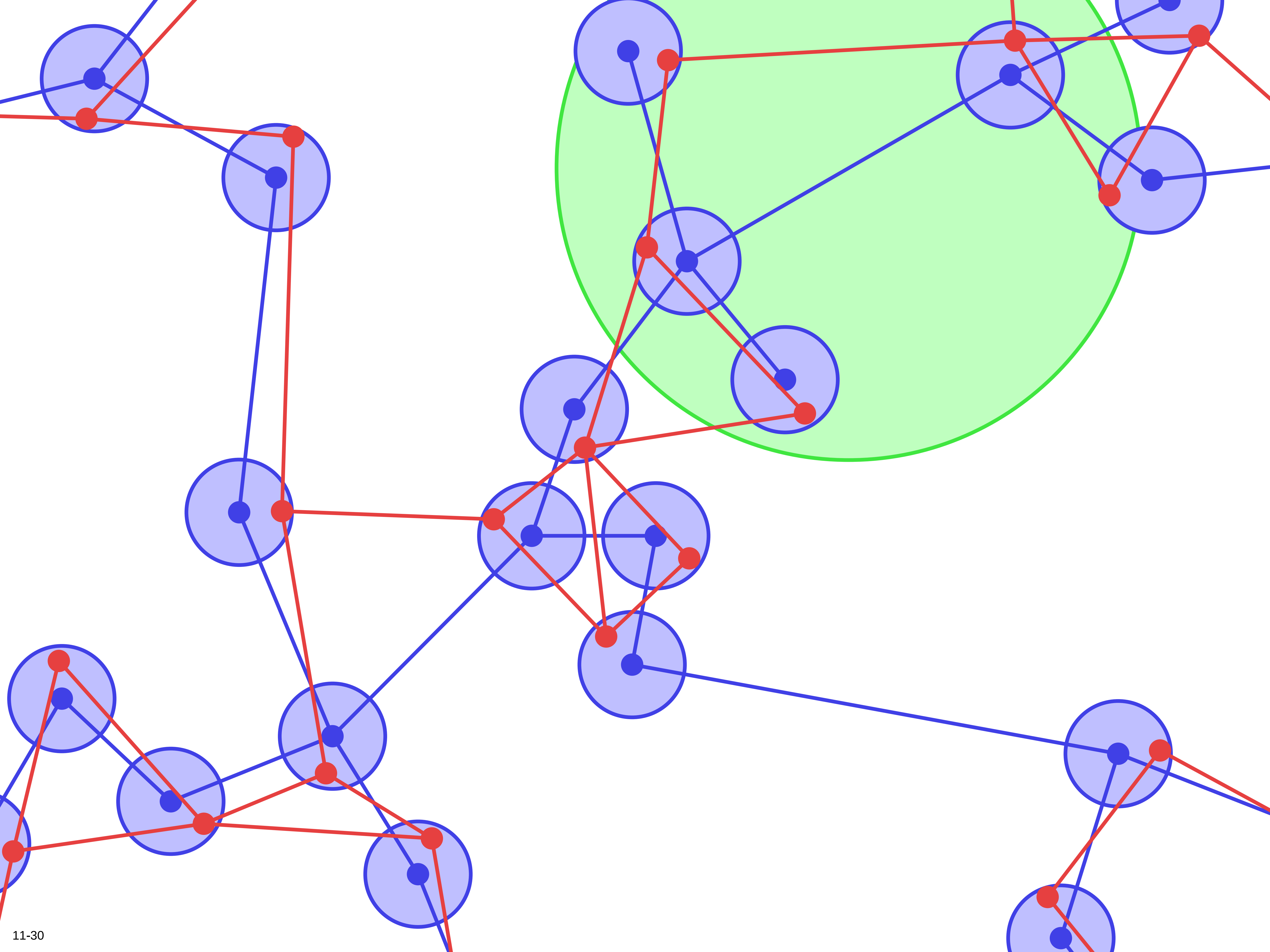
11-26

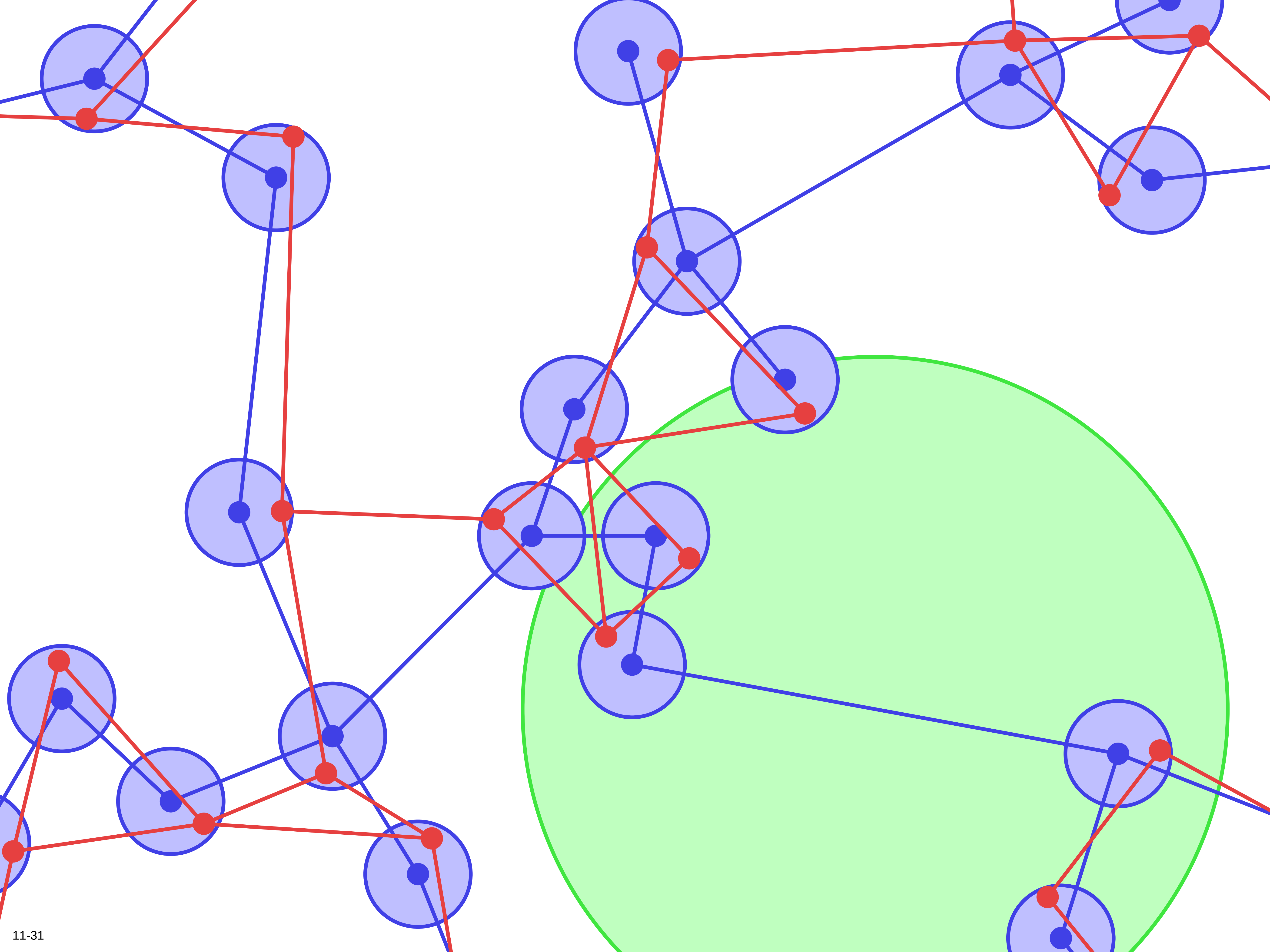




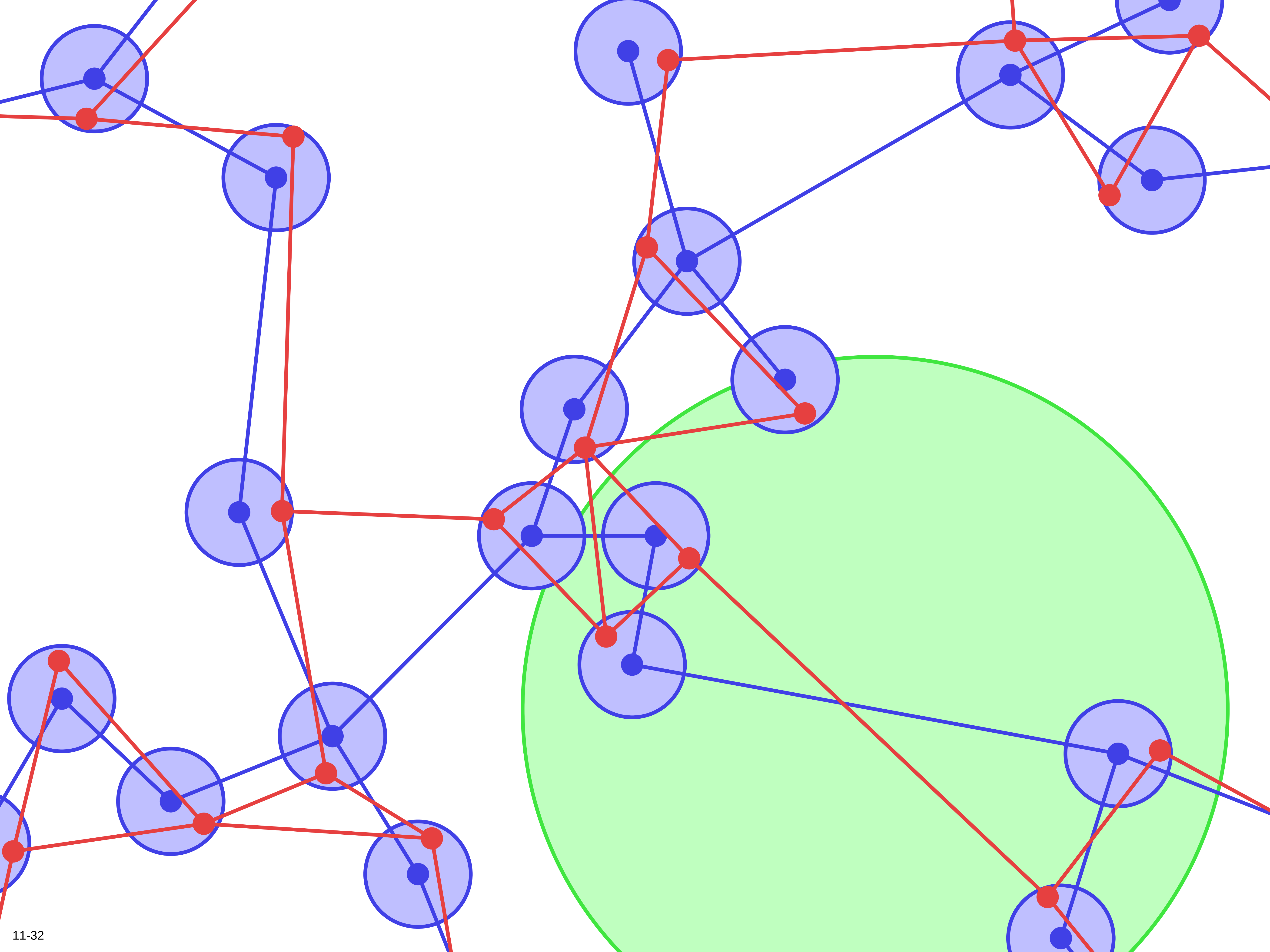


11-29

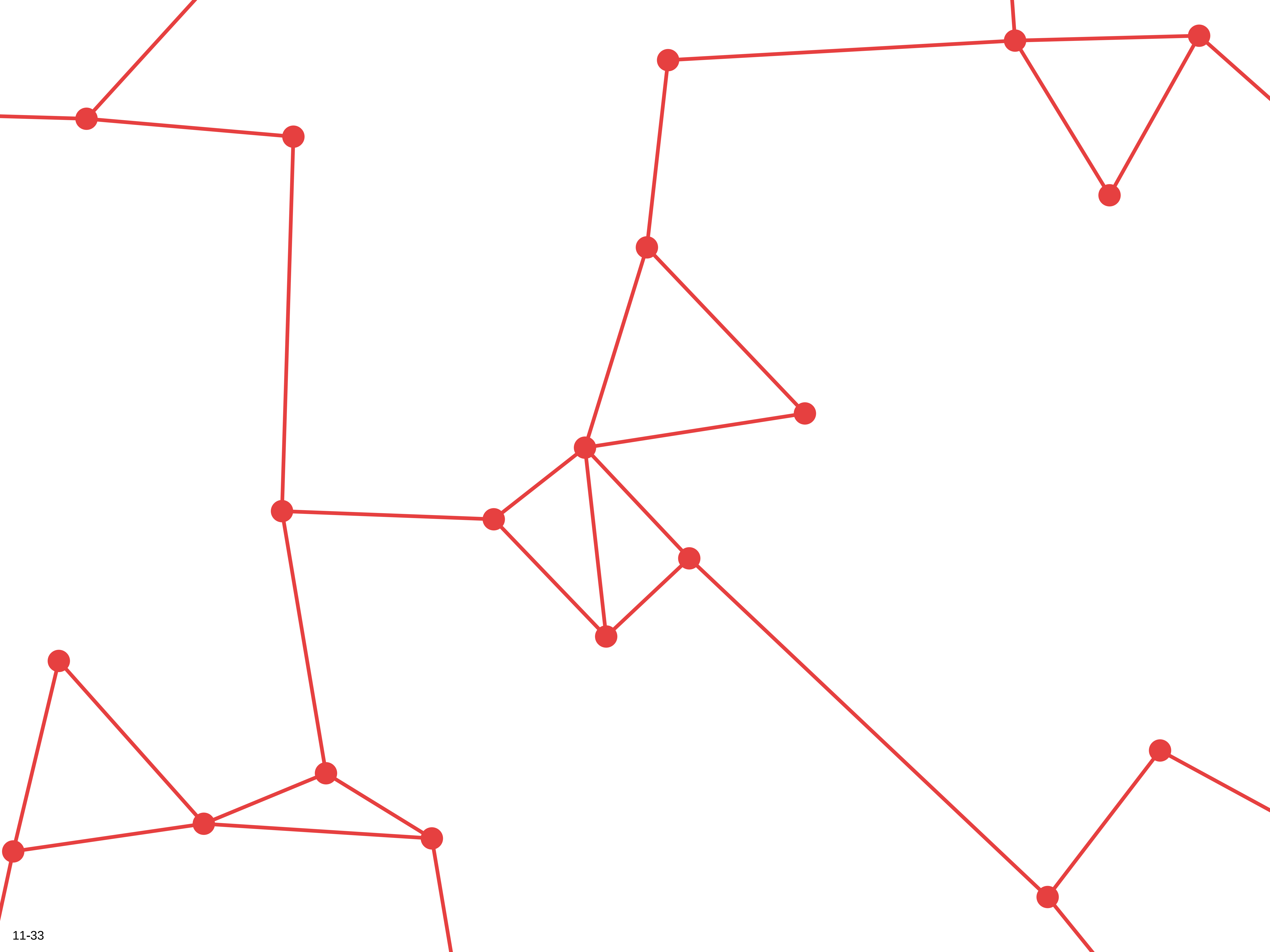


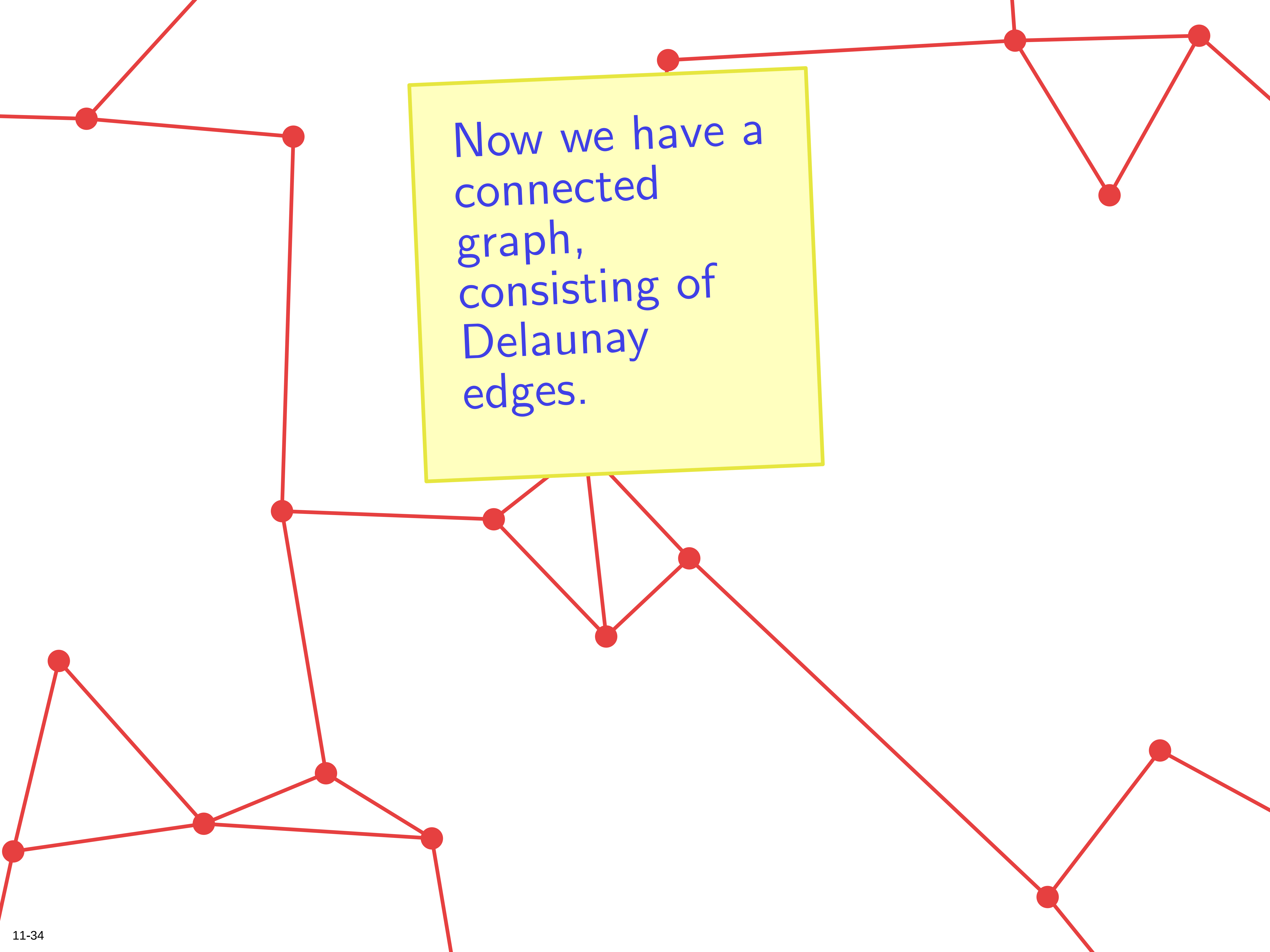


11-31

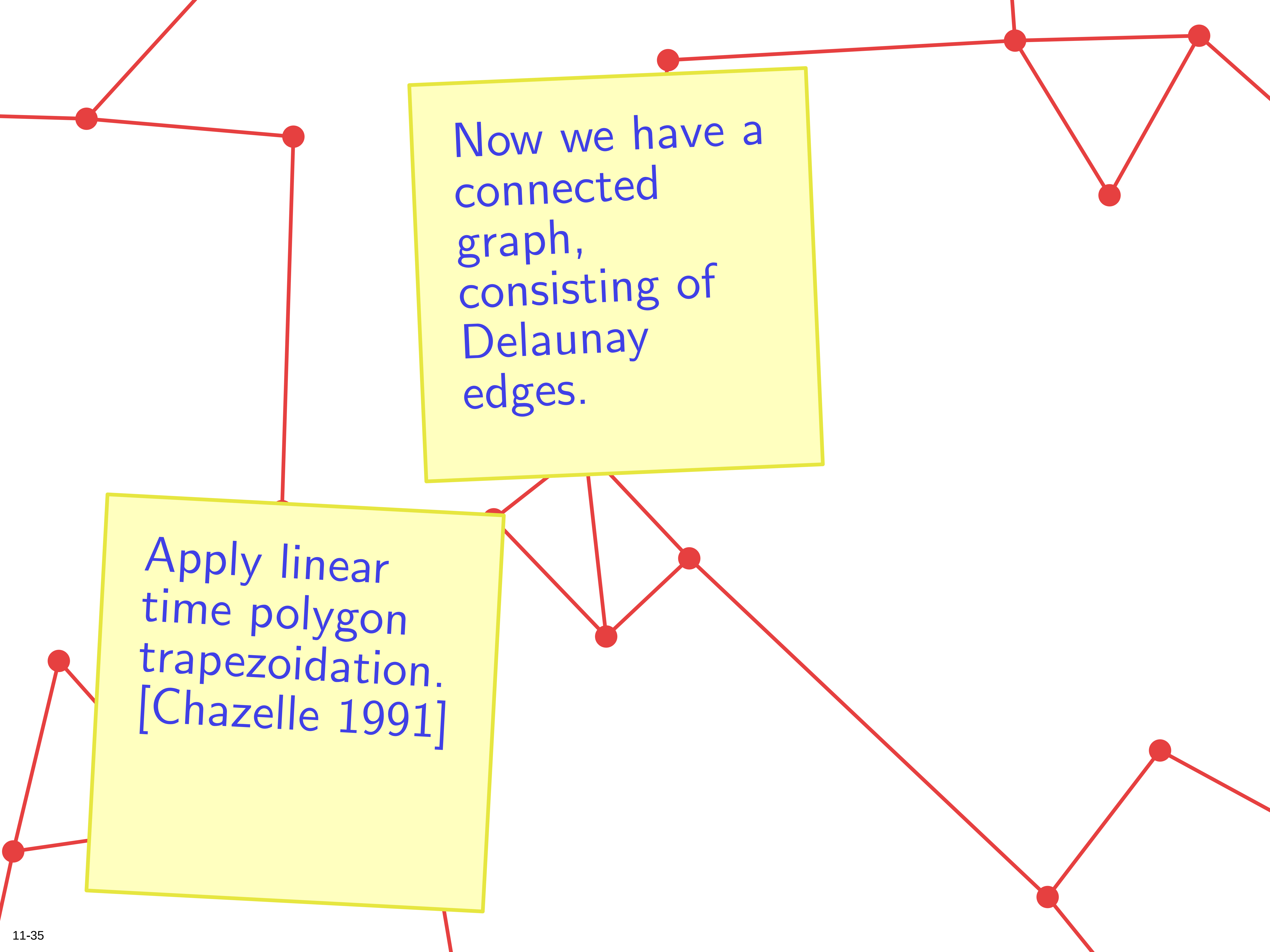


11-32



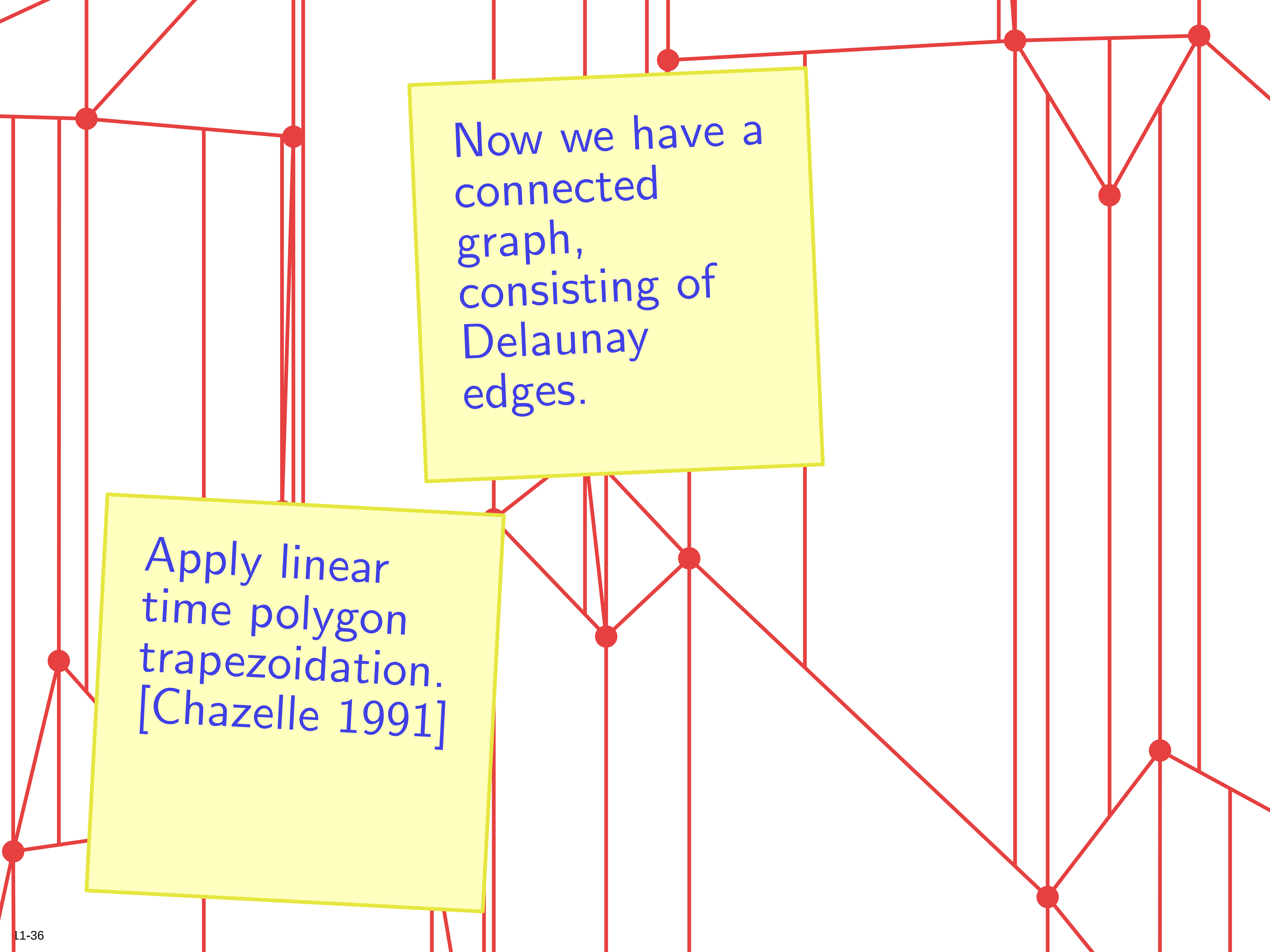


Now we have a connected graph, consisting of Delaunay edges.



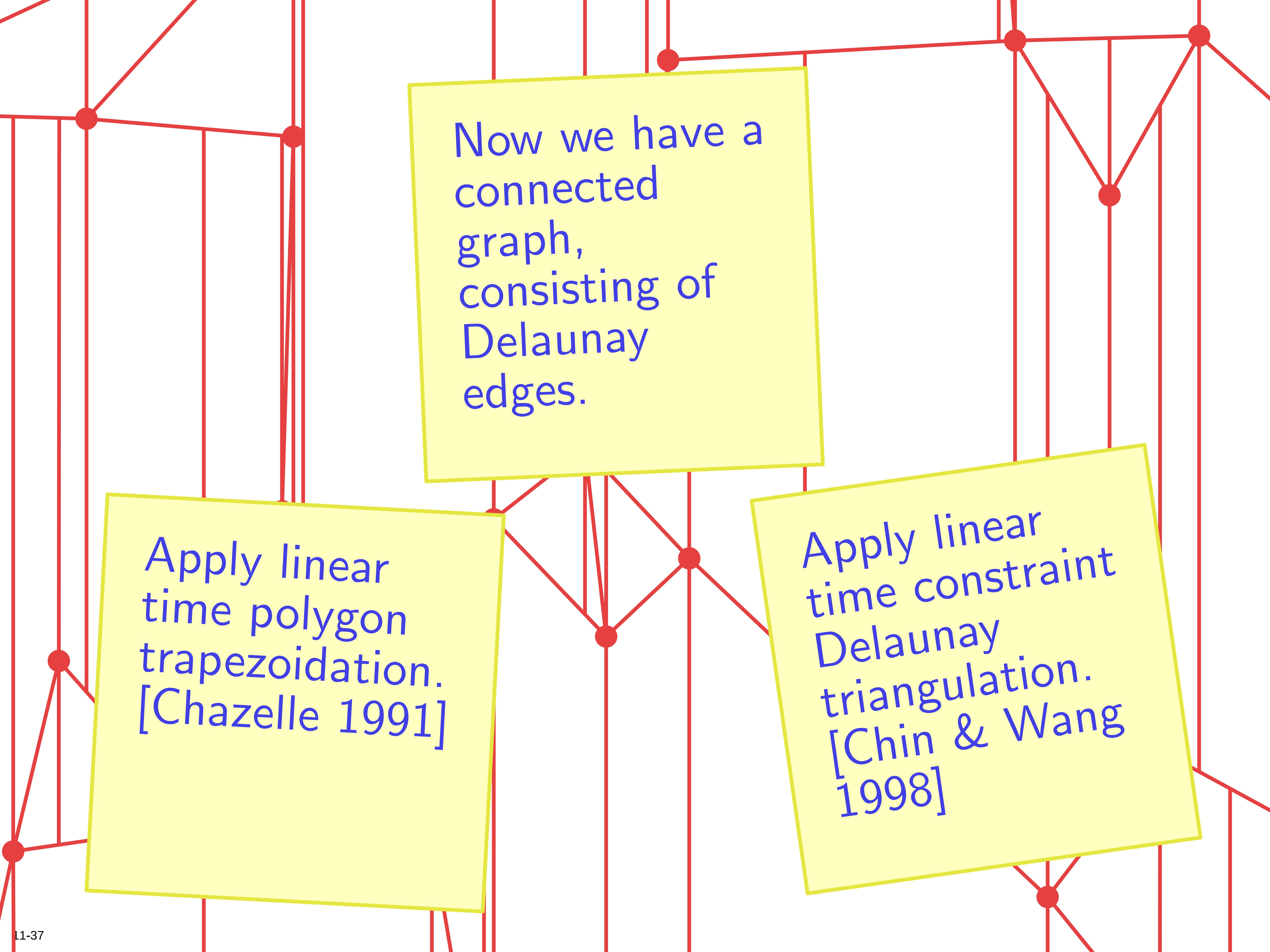
Now we have a connected graph, consisting of Delaunay edges.

Apply linear time polygon trapezoidation. [Chazelle 1991]



Now we have a connected graph, consisting of Delaunay edges.

Apply linear time polygon trapezoidation. [Chazelle 1991]



Now we have a connected graph, consisting of Delaunay edges.

Apply linear time polygon trapezoidation. [Chazelle 1991]

Apply linear time constraint Delaunay triangulation. [Chin & Wang 1998]

Now we have a connected graph, consisting of Delaunay edges.

Apply linear time polygon trapezoidation. [Chazelle 1991]

Apply linear time constraint Delaunay triangulation. [Chin & Wang 1998]

Now we have a connected graph, consisting of Delaunay

Done!

Apply linear time polygon trapezoidation. [Chazelle 1991]

pply linear
me constraint
elaunay
riangulation.
Chin & Wang
1998]

Time to conclude.

We preprocess a set of imprecise points in $O(n \log n)$ time.

We preprocess a set of imprecise points in $O(n \log n)$ time.

Given an exact sample, we compute the Delaunay triangulation in $O(n)$ time.

We preprocess a set of imprecise points in $O(n \log n)$ time.

Given an exact sample, we compute the Delaunay triangulation in $O(n)$ time.

Can be extended to partially overlapping discs, or other shapes.

We preprocess a set of imprecise points in $O(n \log n)$ time.

Given an exact sample, we compute the Delaunay triangulation in $O(n)$ time.

Can be extended to partially overlapping discs, or other shapes.

Future work: can we compute more structure, to avoid using Chin & Wang?

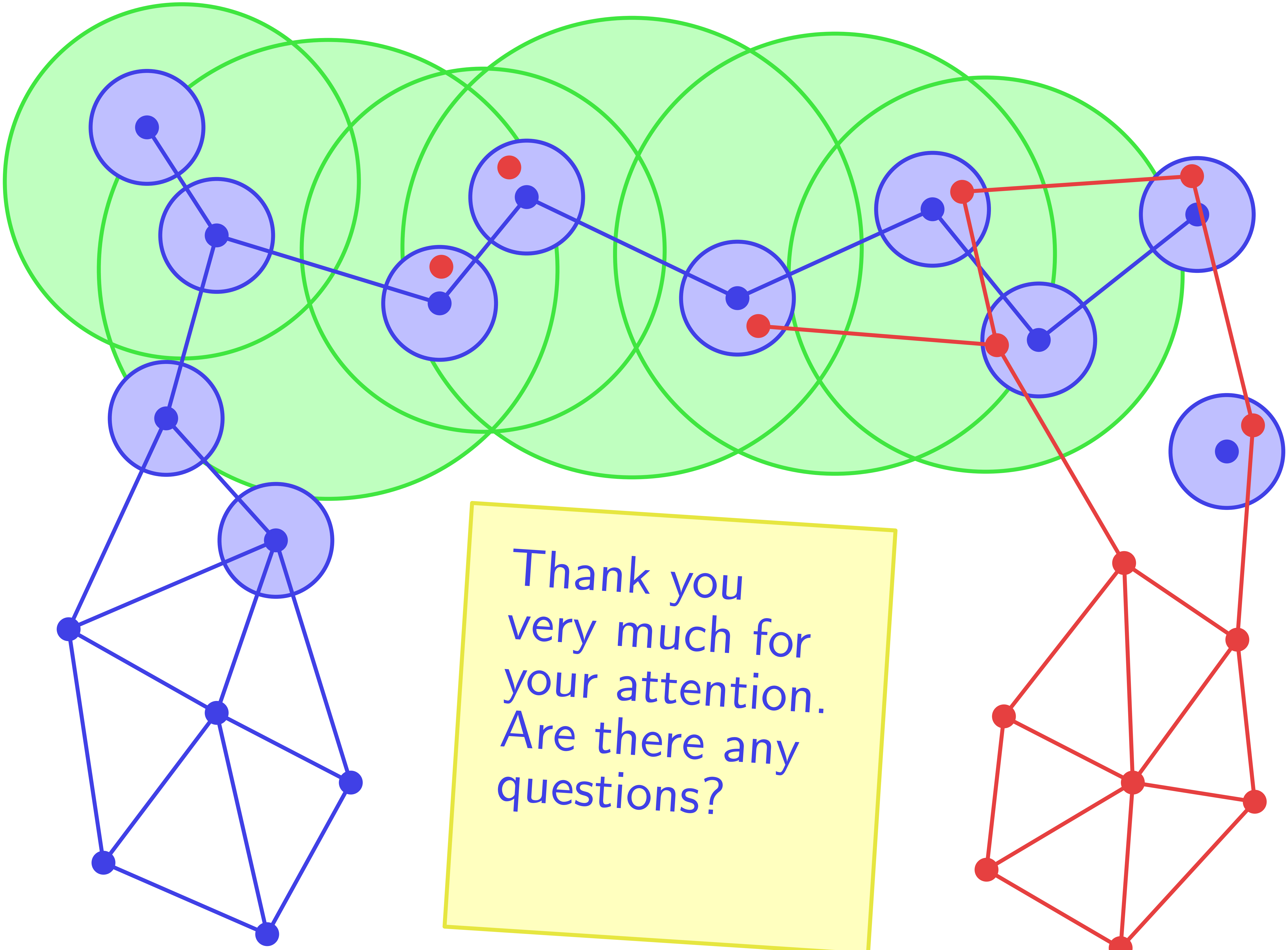
We preprocess a set of imprecise points in $O(n \log n)$ time.

Given an exact sample, we compute the Delaunay triangulation in $O(n)$ time.

Can be extended to partially overlapping discs, or other shapes.

Future work: can we compute more structure, to avoid using Chin & Wang?

Future work: can our techniques be applied in settings with moving points?


Thank you
very much for
your attention.
Are there any
questions?