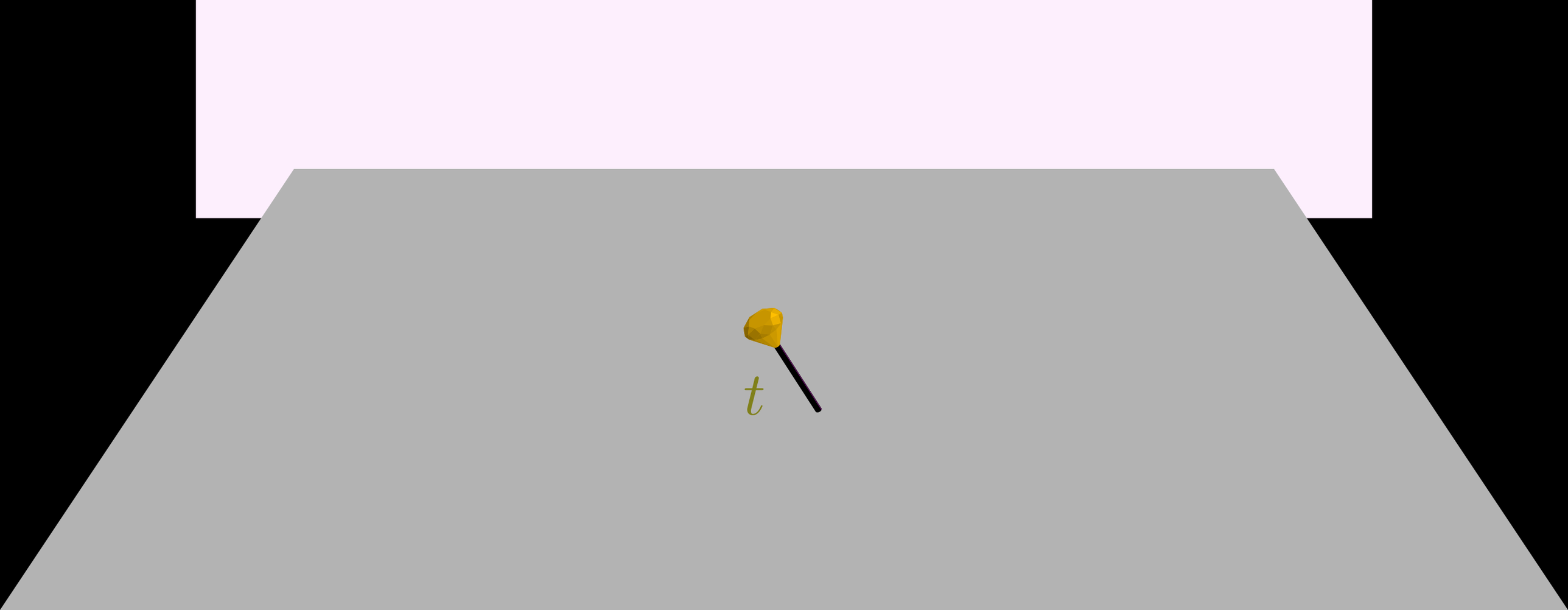
ON THE COMPLEXITY OF BARRIER RESILIENCE FOR FAT REGIONS

Matias Korman
Maarten Löffler
Rodrigo Silveira
Darren Strash

INTRODUCTION

Let t be a desirable point in the plane.

Let t be a desirable point in the plane.

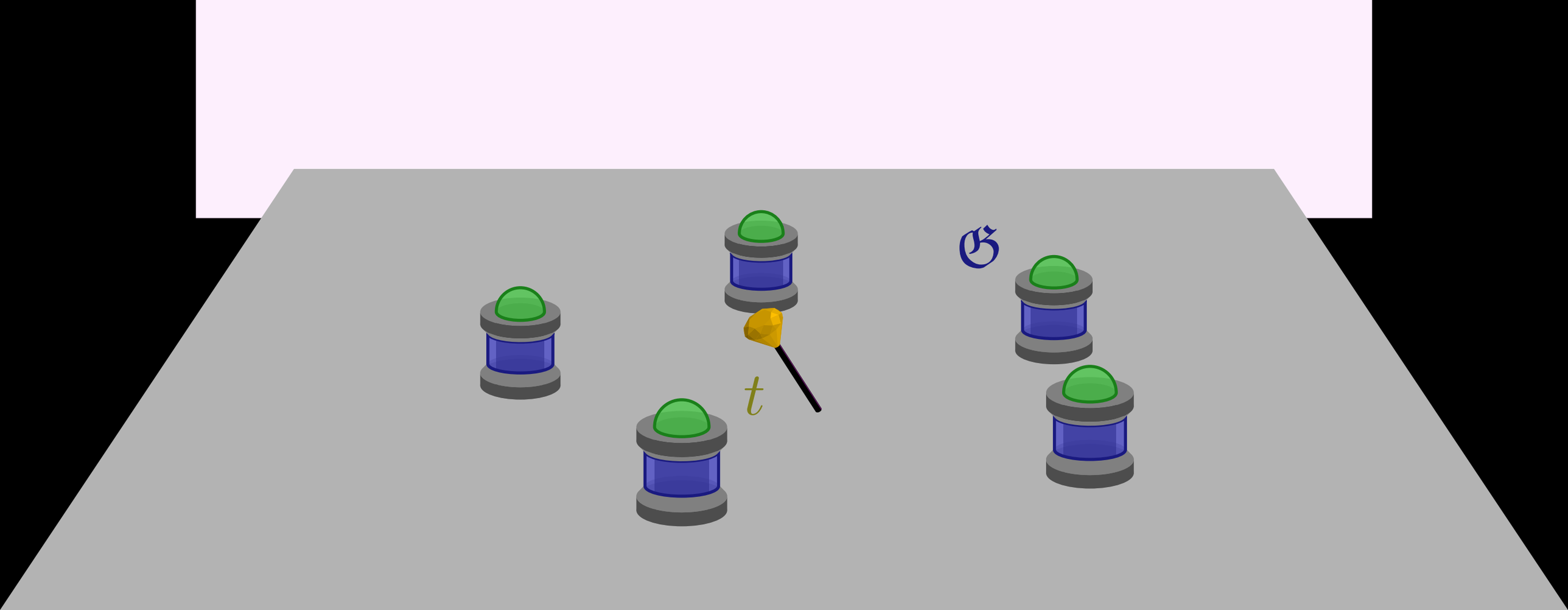


Let t be a desirable point in the plane.

Let $\mathfrak{G}$ be a set of guards.

Let t be a desirable point in the plane.

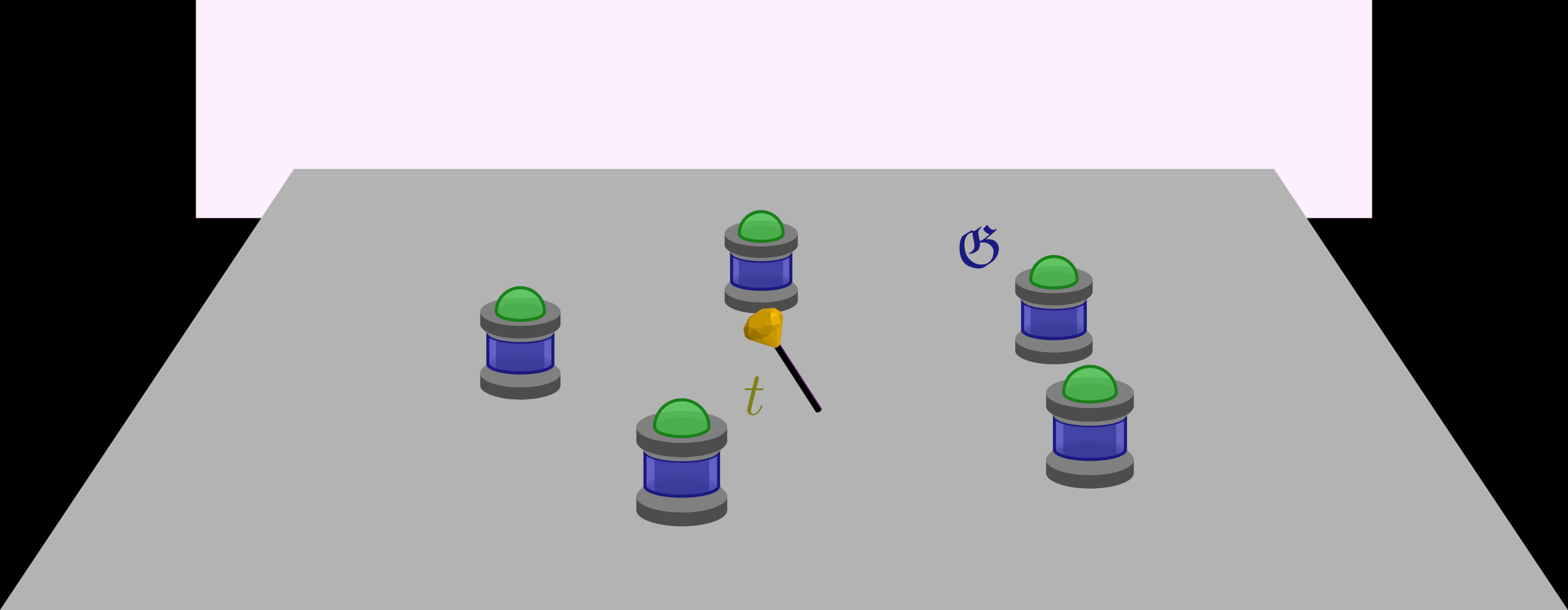
Let $\mathfrak{G}$ be a set of guards.



Let t be a desirable point in the plane.

Let $\mathfrak{G}$ be a set of guards.

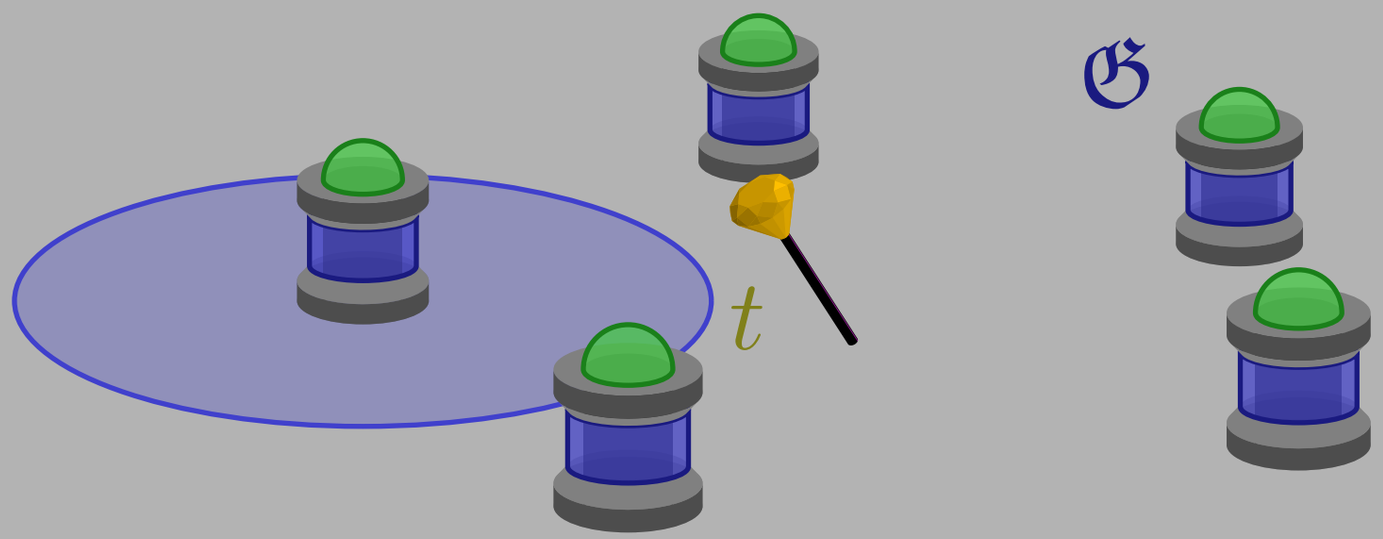
Each guard can detect movement in a given *region*.



Let t be a desirable point in the plane.

Let $\mathfrak{G}$ be a set of guards.

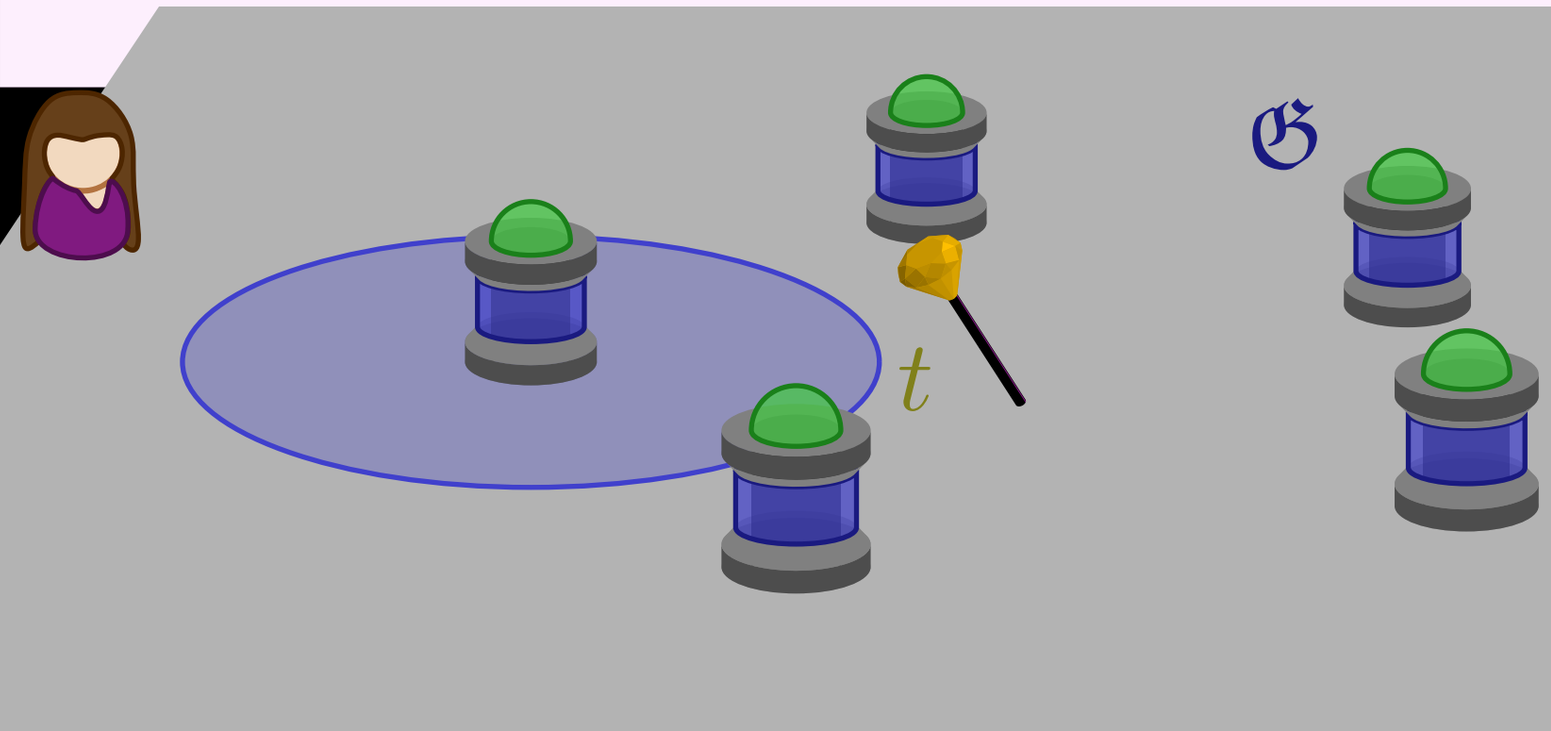
Each guard can detect movement in a given *region*.



Let t be a desirable point in the plane.

Let $\mathfrak{G}$ be a set of guards.

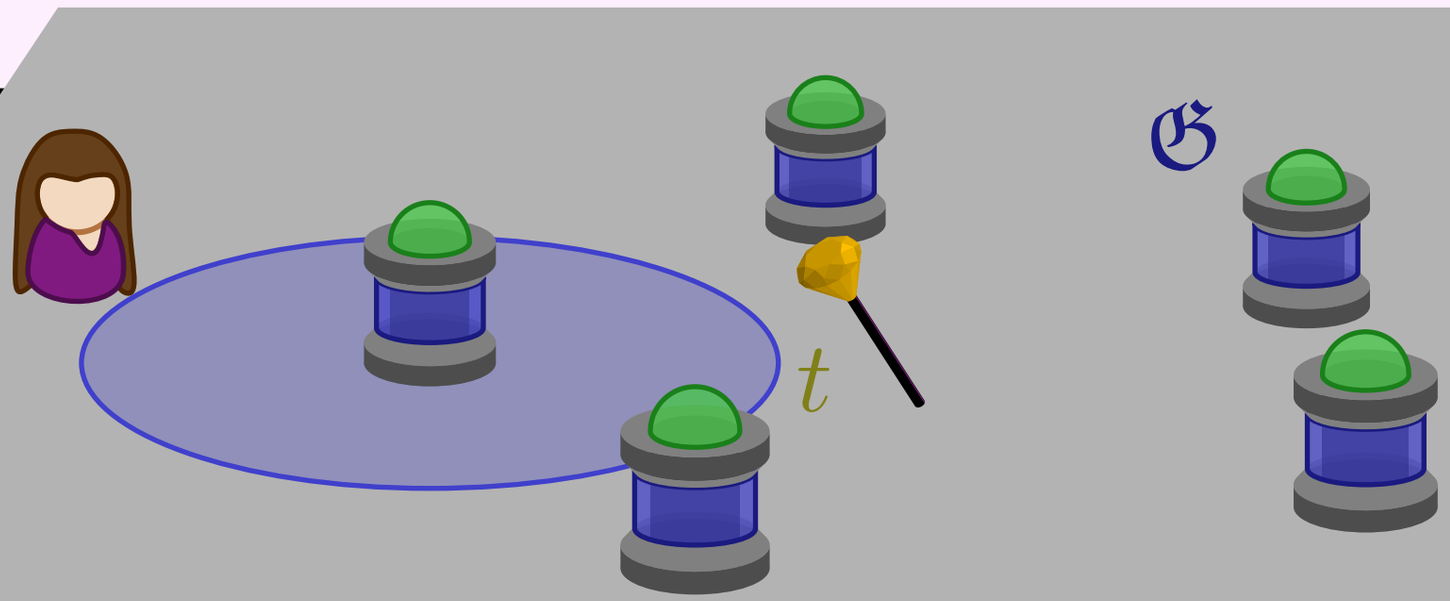
Each guard can detect movement in a given *region*.



Let t be a desirable point in the plane.

Let $\mathfrak{G}$ be a set of guards.

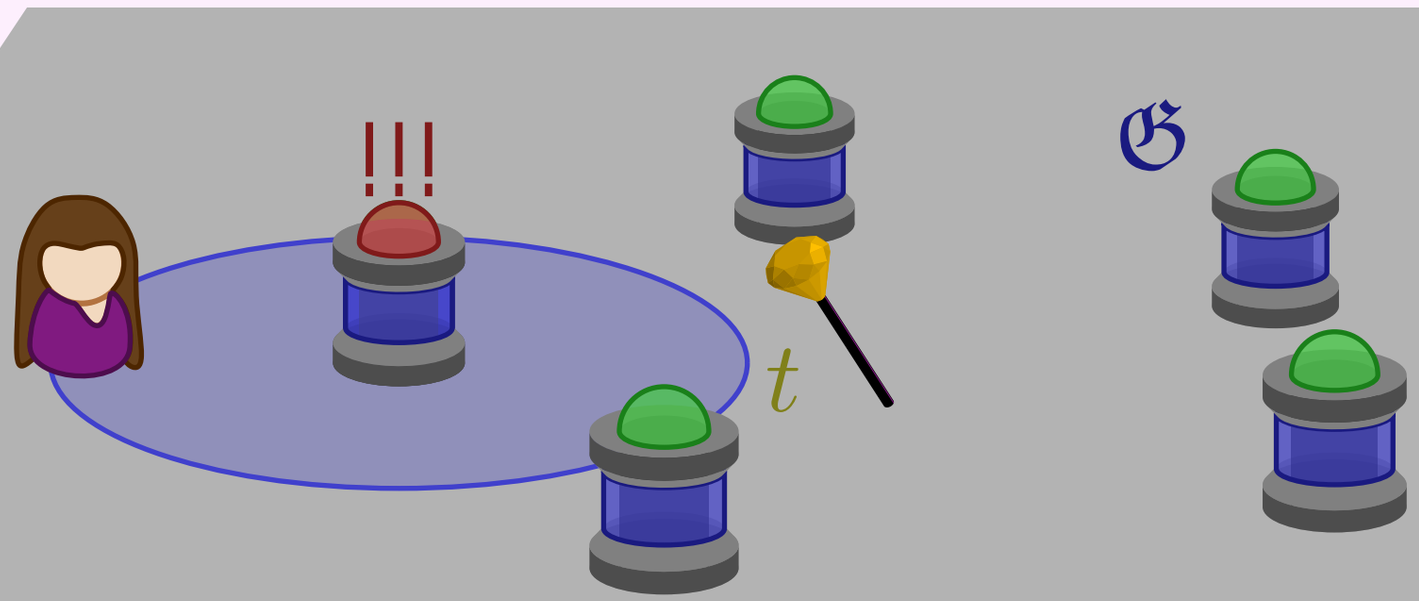
Each guard can detect movement in a given *region*.



Let t be a desirable point in the plane.

Let $\mathfrak{G}$ be a set of guards.

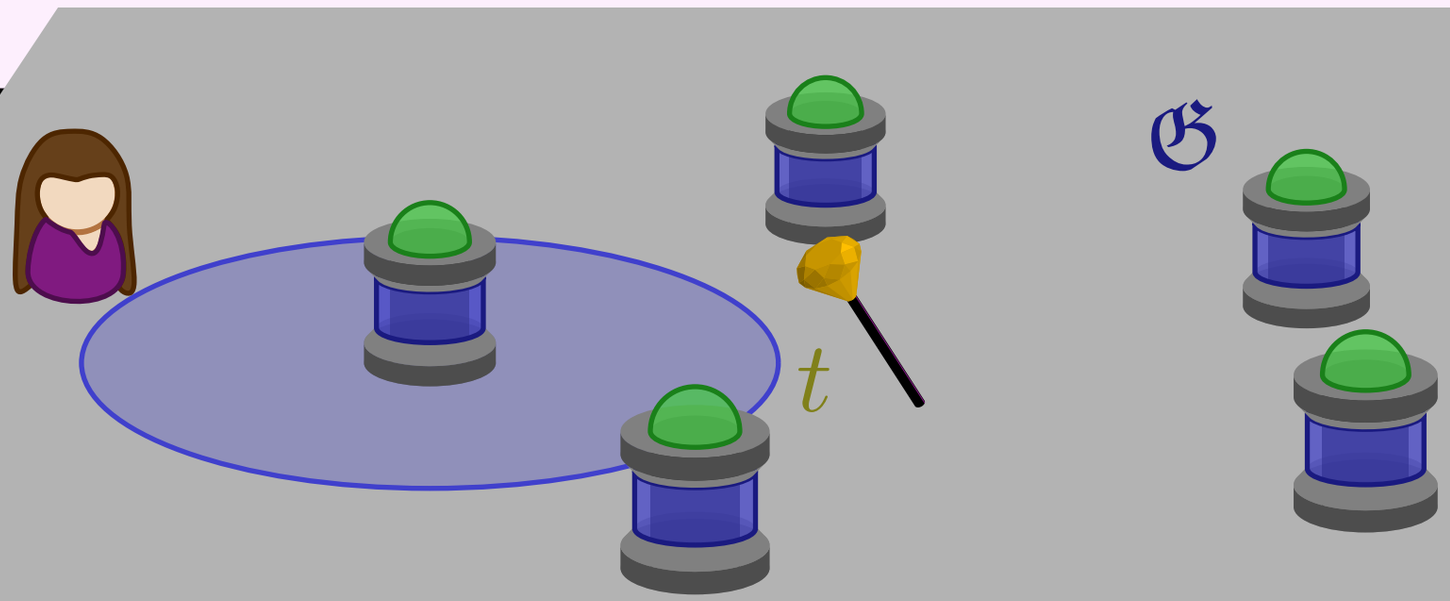
Each guard can detect movement in a given *region*.



Let t be a desirable point in the plane.

Let $\mathfrak{G}$ be a set of guards.

Each guard can detect movement in a given *region*.

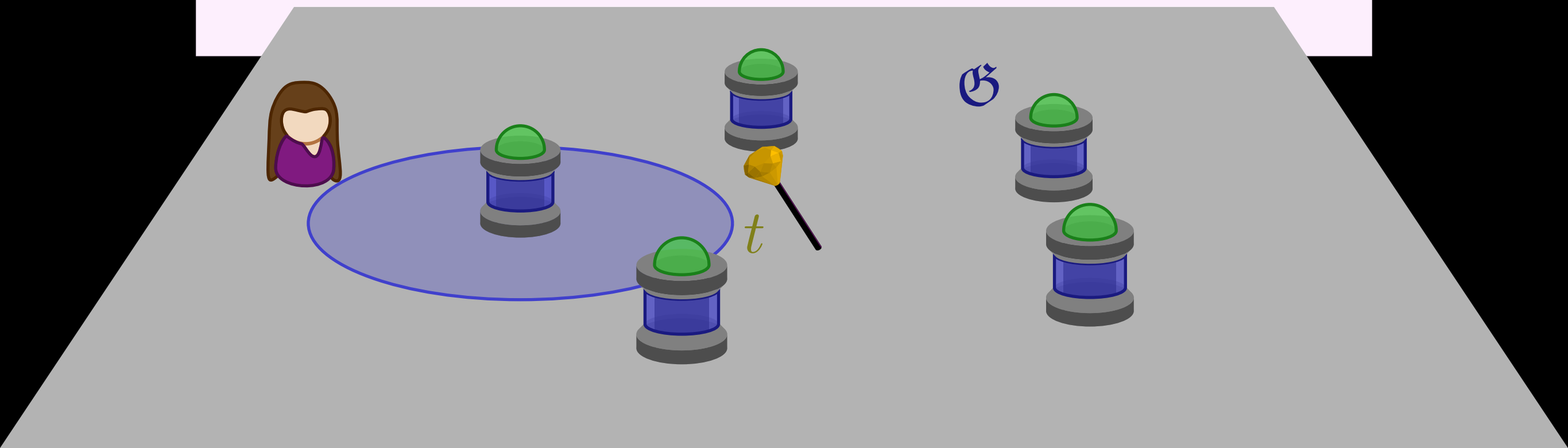


Let t be a desirable point in the plane.

Let $\mathfrak{G}$ be a set of guards.

Each guard can detect movement in a given *region*.

Let $\mathcal{R}$ be the set of guarded regions.

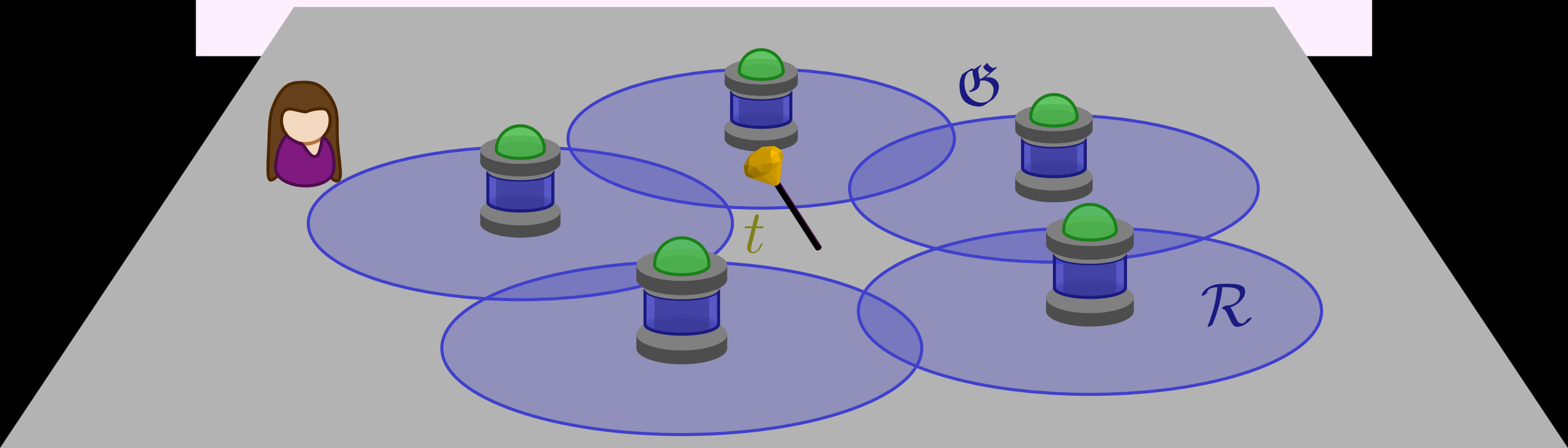


Let t be a desirable point in the plane.

Let $\mathfrak{G}$ be a set of guards.

Each guard can detect movement in a given *region*.

Let $\mathcal{R}$ be the set of guarded regions.



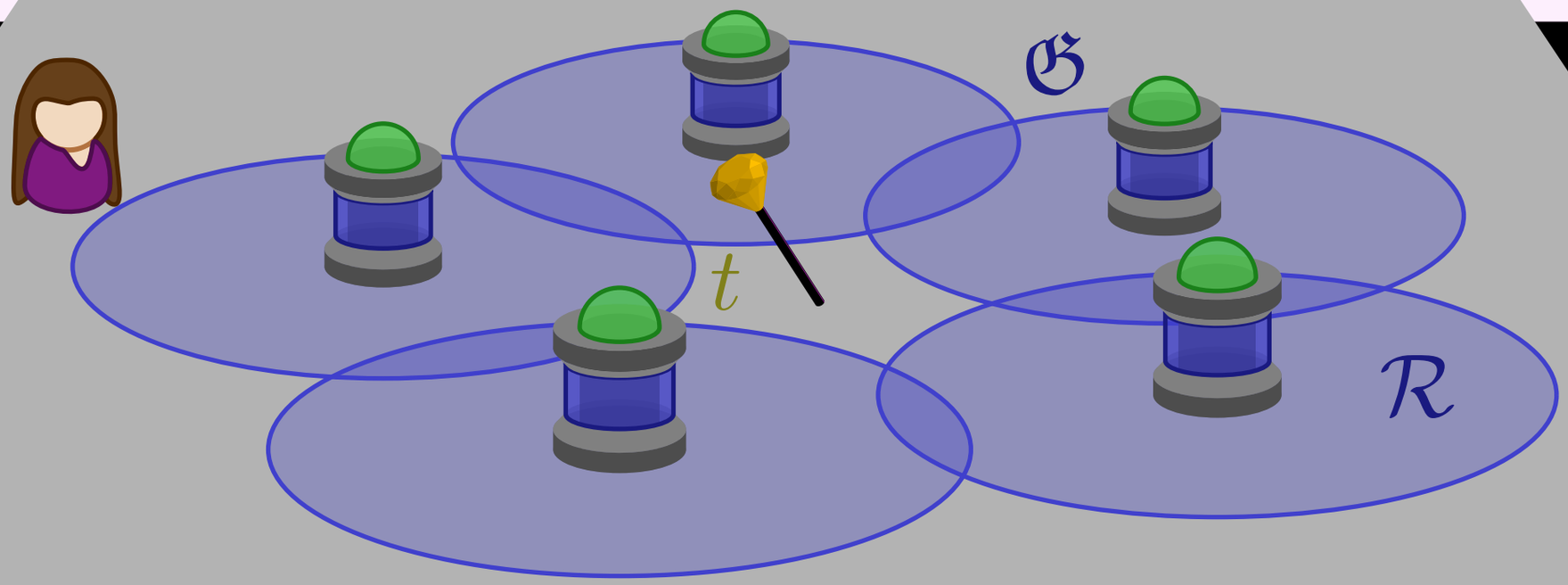
Let t be a desirable point in the plane.

Let $\mathfrak{G}$ be a set of guards.

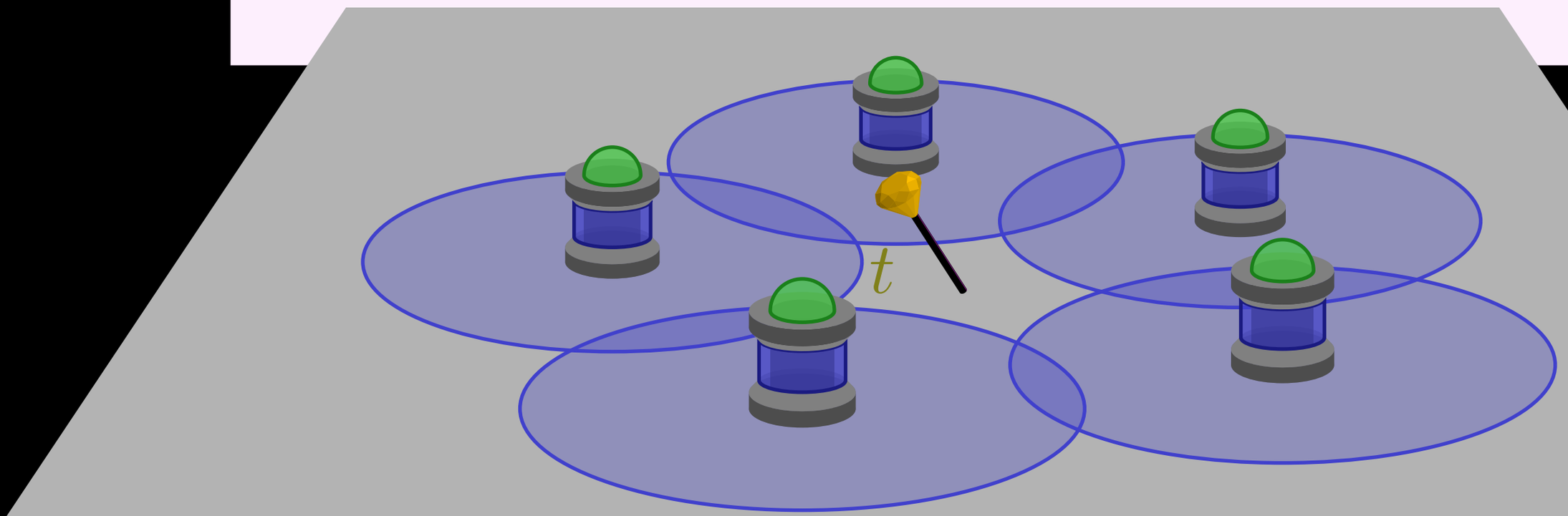
Each guard can detect movement in a given *region*.

Let $\mathcal{R}$ be the set of guarded regions.

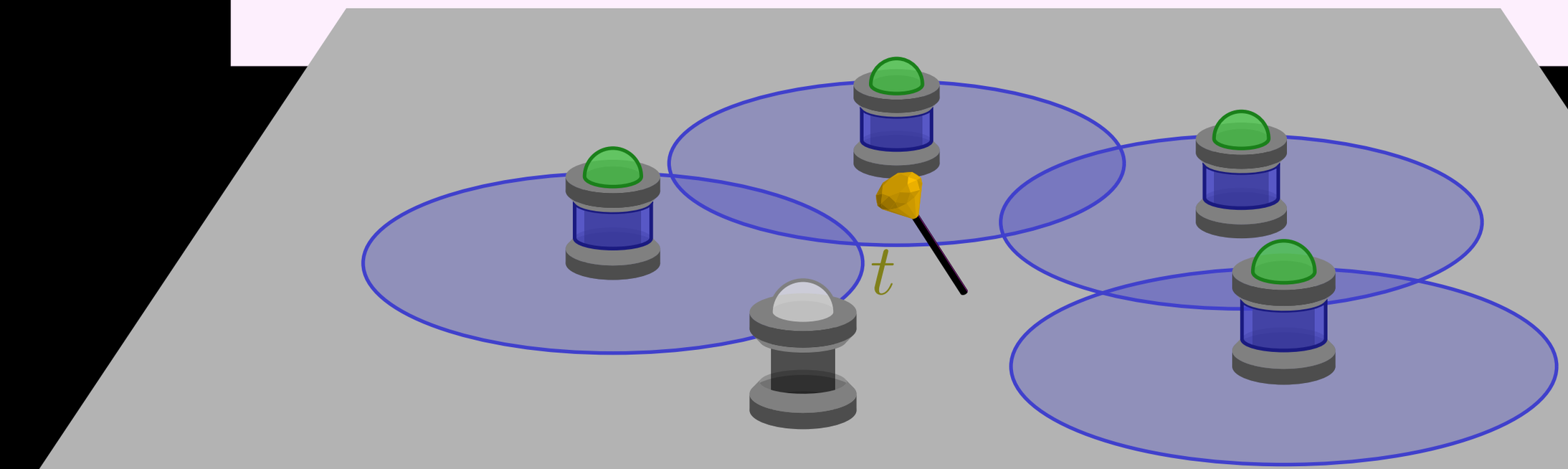
We say $\mathcal{R}$ *guards* t if every path to t passes through one of the regions.



Sometimes, guards fail.

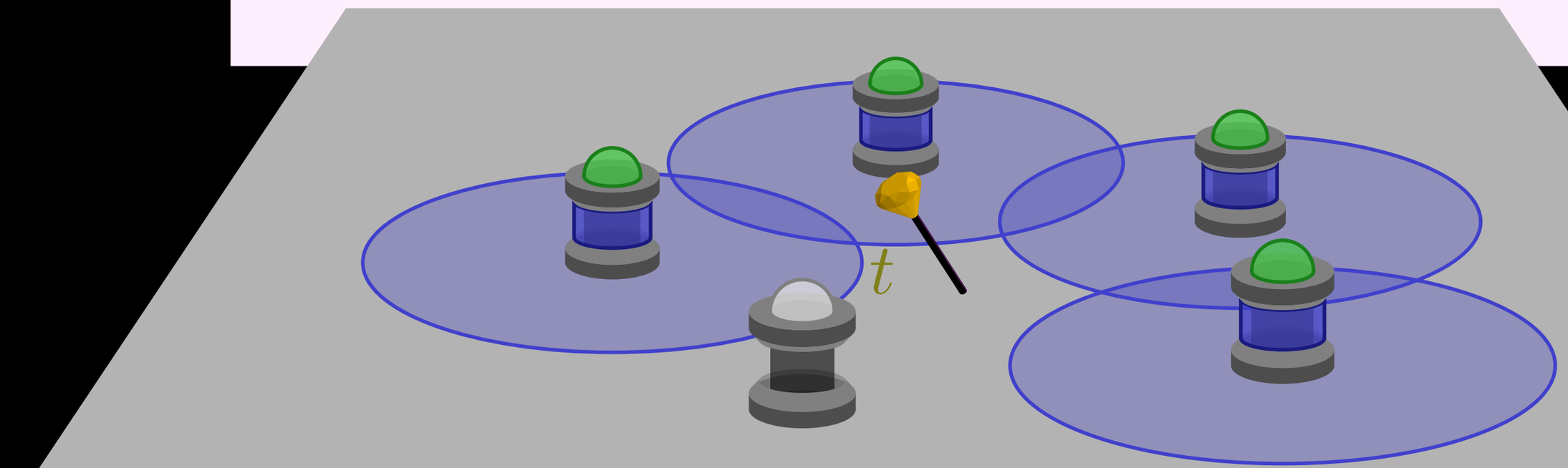


Sometimes, guards fail.



Sometimes, guards fail.

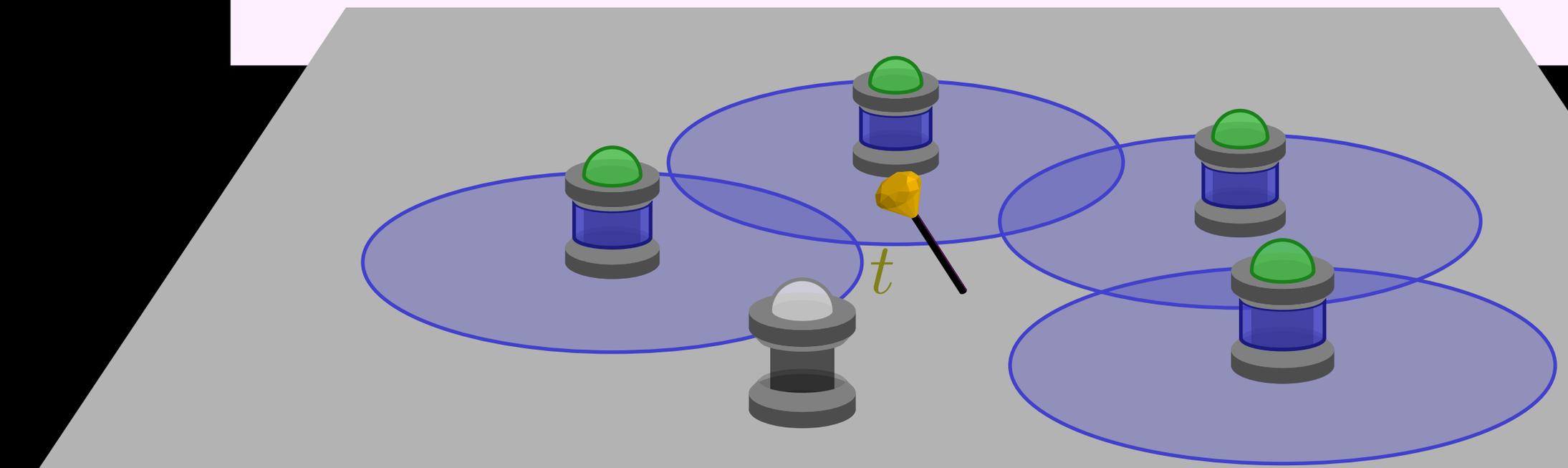
When they do, t may no longer be guarded!



Sometimes, guards fail.

When they do, t may no longer be guarded!

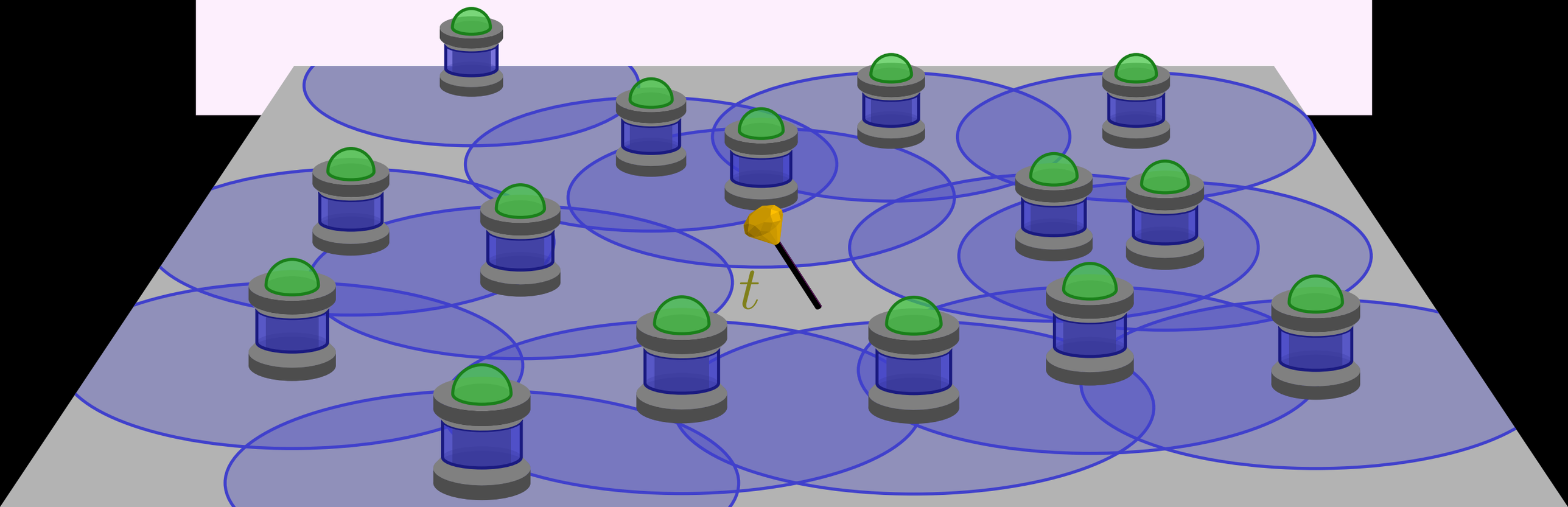
The *resilience* of $\mathcal{R}$ is the smallest number of guards that need to fail before $\mathcal{R}$ no longer guards t.



Sometimes, guards fail.

When they do, t may no longer be guarded!

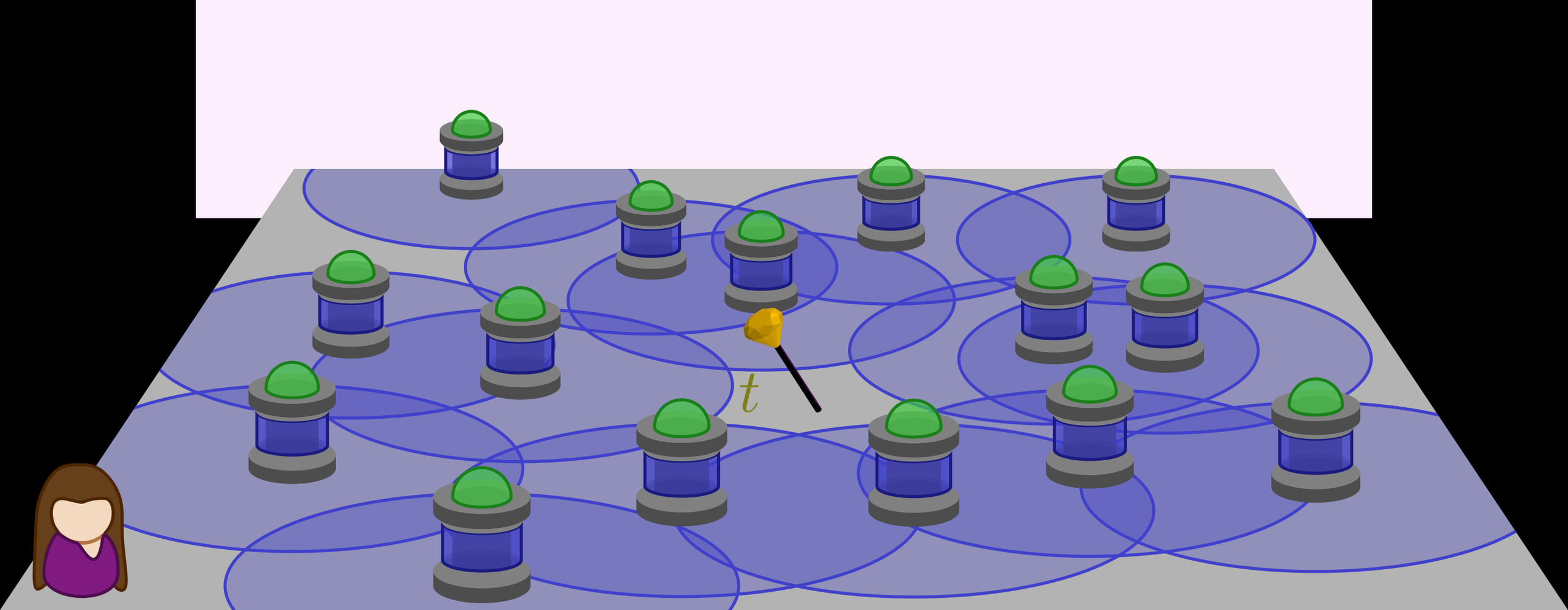
The *resilience* of $\mathcal{R}$ is the smallest number of guards that need to fail before $\mathcal{R}$ no longer guards t.



Sometimes, guards fail.

When they do, t may no longer be guarded!

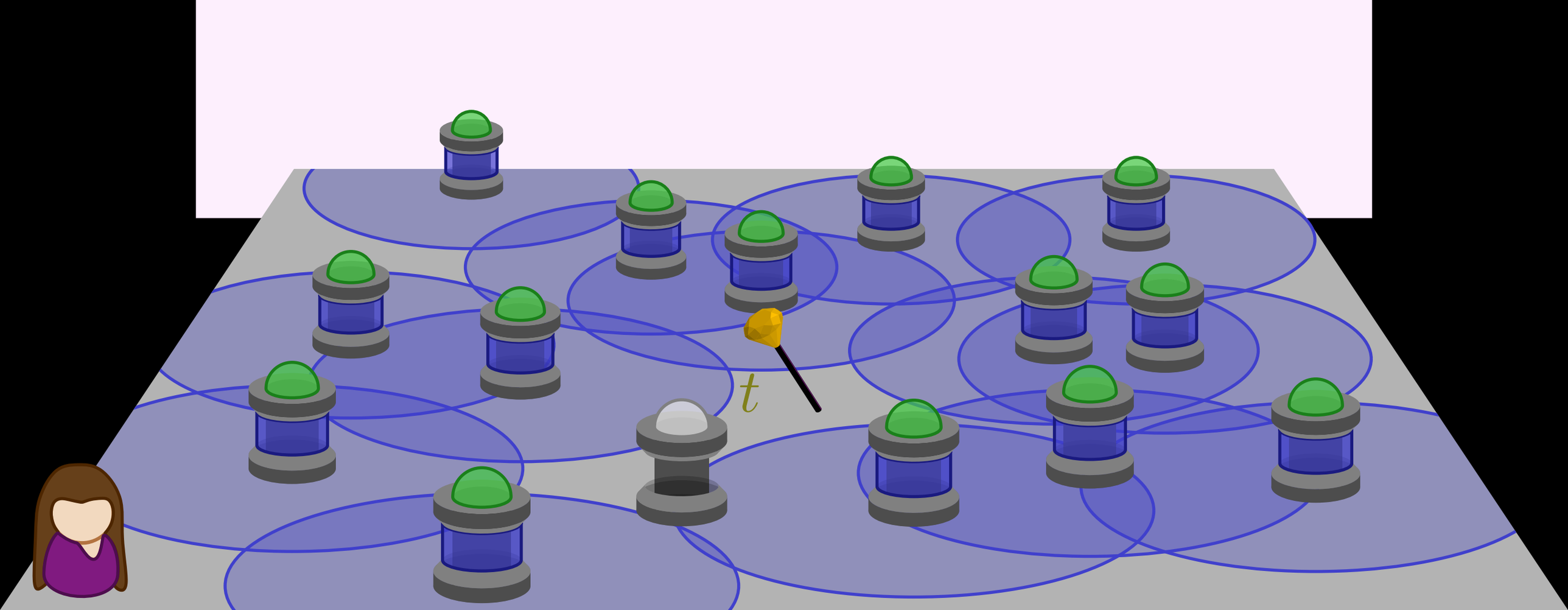
The *resilience* of $\mathcal{R}$ is the smallest number of guards that need to fail before $\mathcal{R}$ no longer guards t.



Sometimes, guards fail.

When they do, t may no longer be guarded!

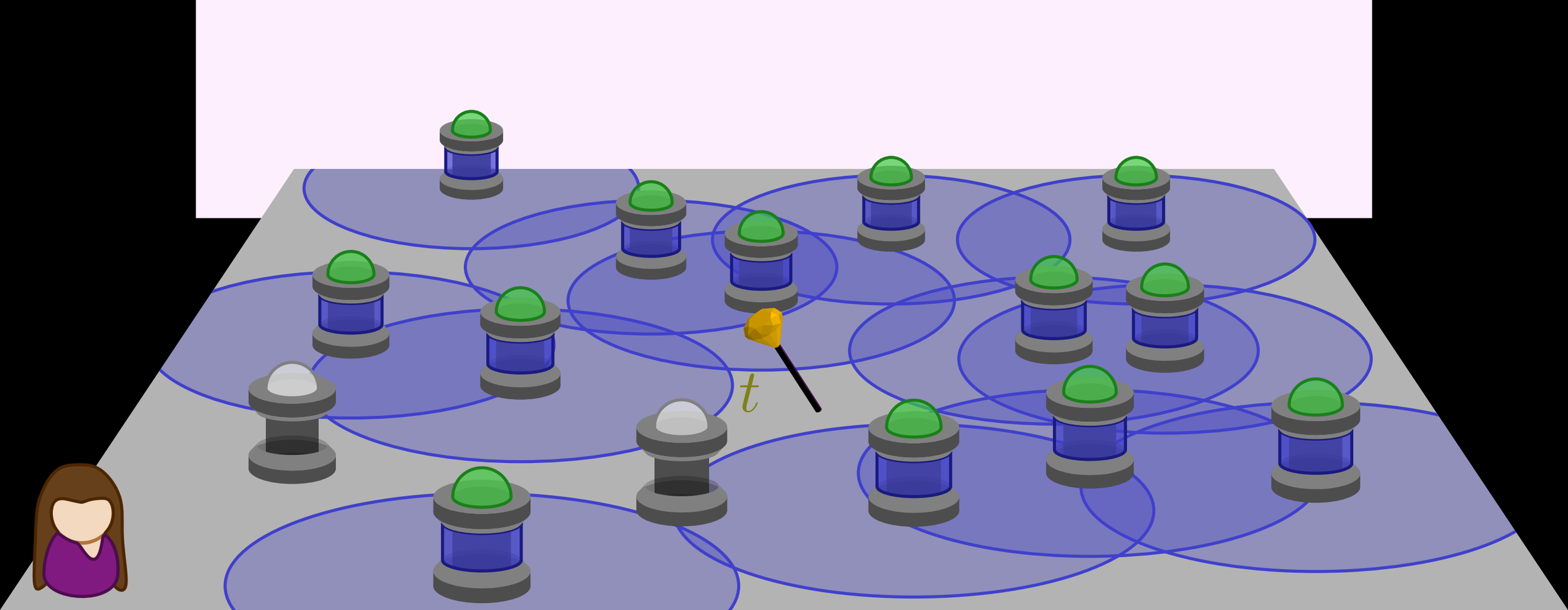
The *resilience* of $\mathcal{R}$ is the smallest number of guards that need to fail before $\mathcal{R}$ no longer guards t.



Sometimes, guards fail.

When they do, t may no longer be guarded!

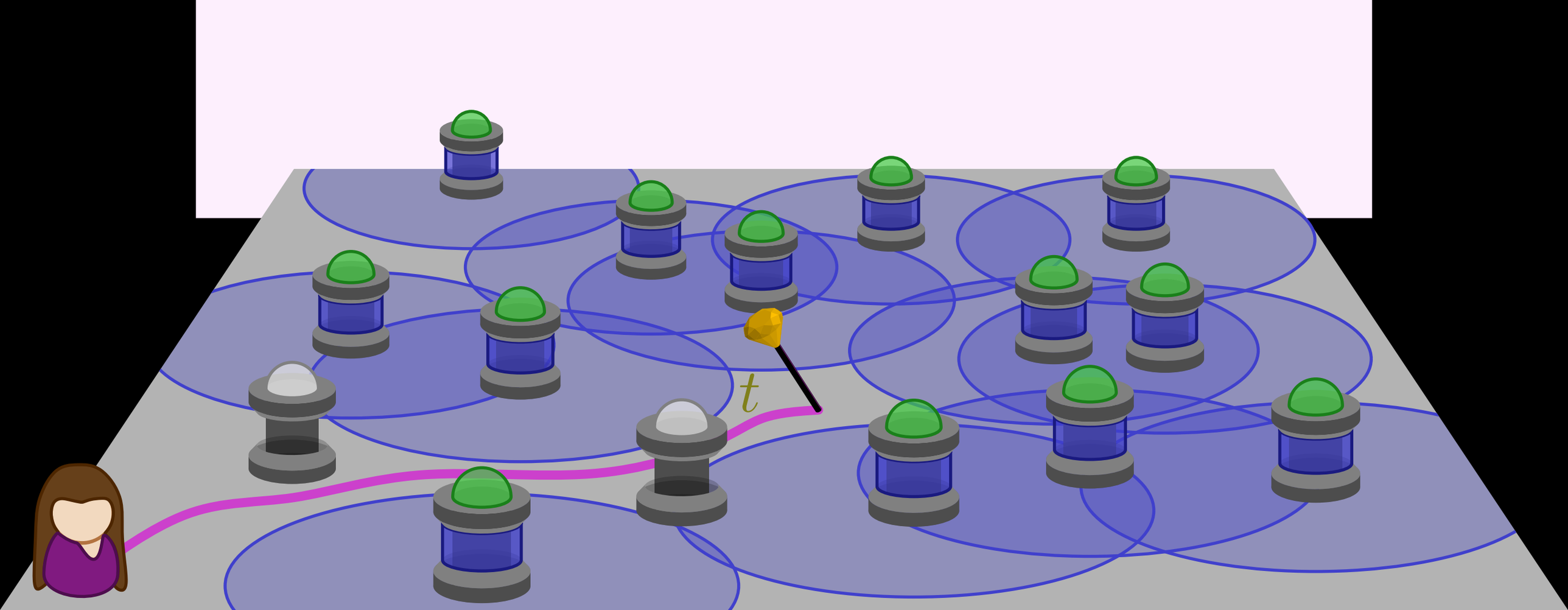
The *resilience* of $\mathcal{R}$ is the smallest number of guards that need to fail before $\mathcal{R}$ no longer guards t.



Sometimes, guards fail.

When they do, t may no longer be guarded!

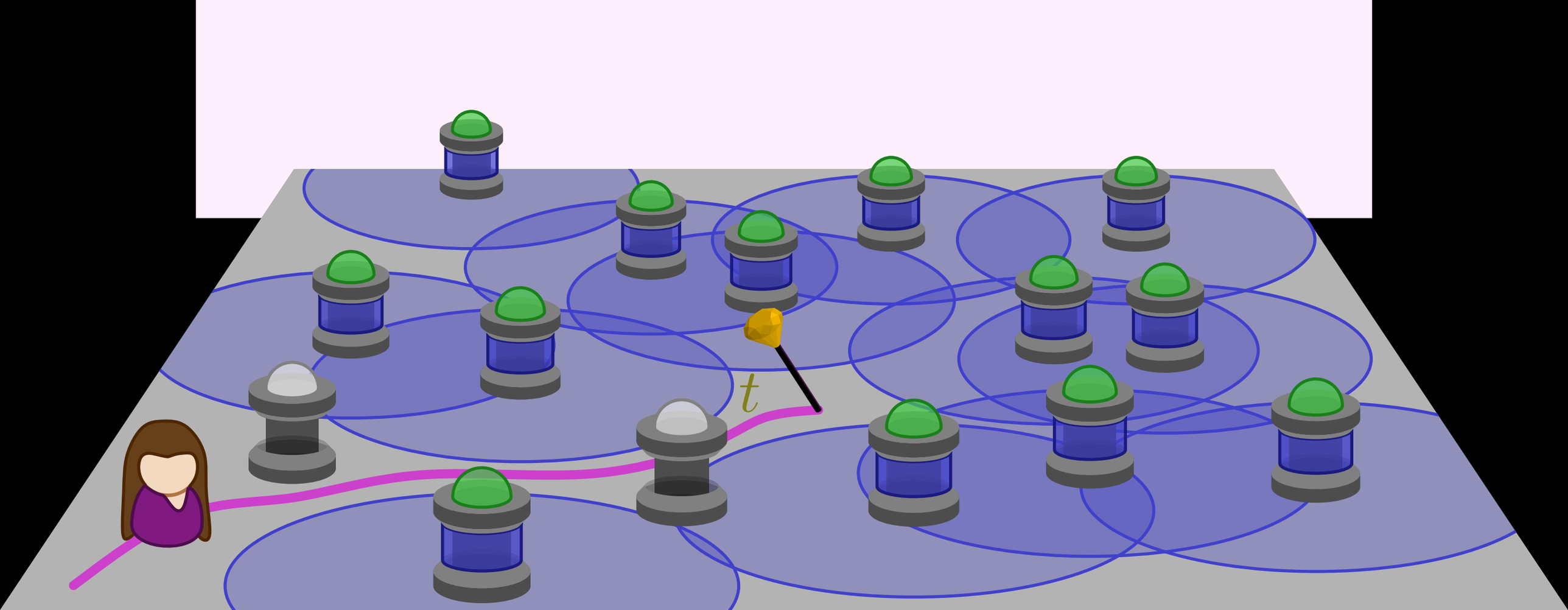
The *resilience* of $\mathcal{R}$ is the smallest number of guards that need to fail before $\mathcal{R}$ no longer guards t.



Sometimes, guards fail.

When they do, t may no longer be guarded!

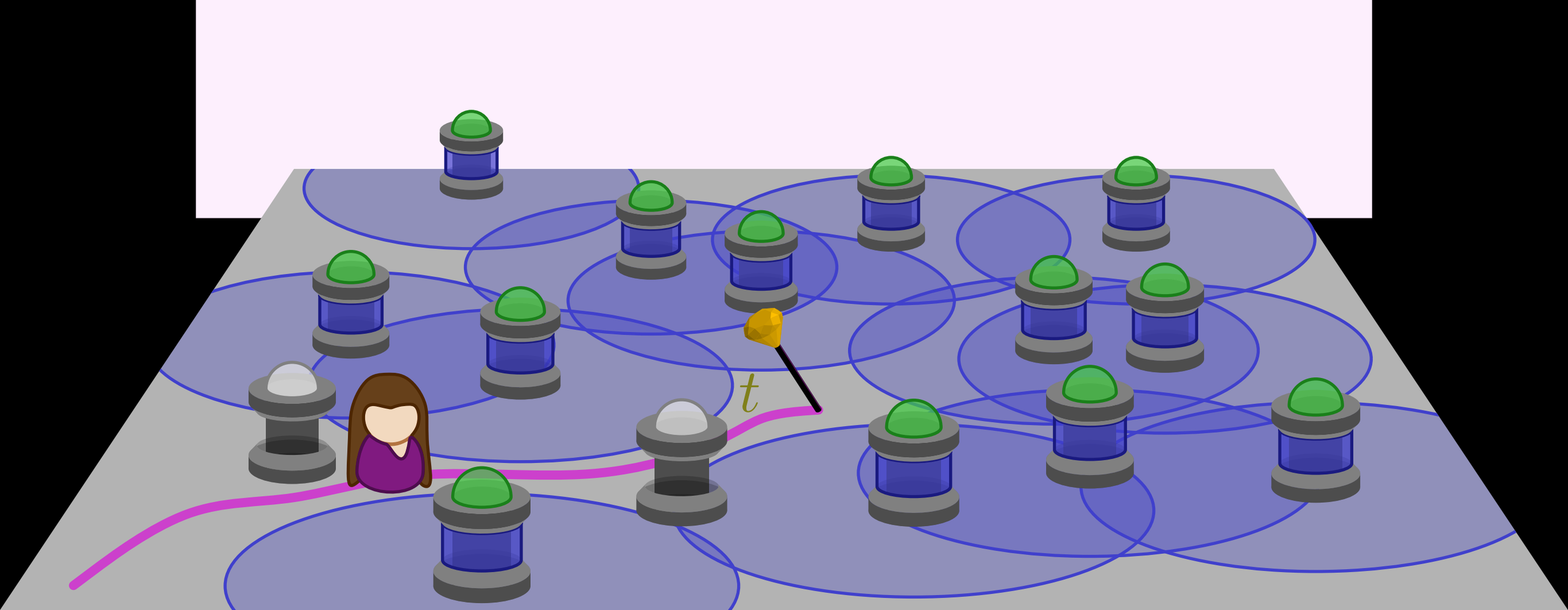
The *resilience* of $\mathcal{R}$ is the smallest number of guards that need to fail before $\mathcal{R}$ no longer guards t.



Sometimes, guards fail.

When they do, t may no longer be guarded!

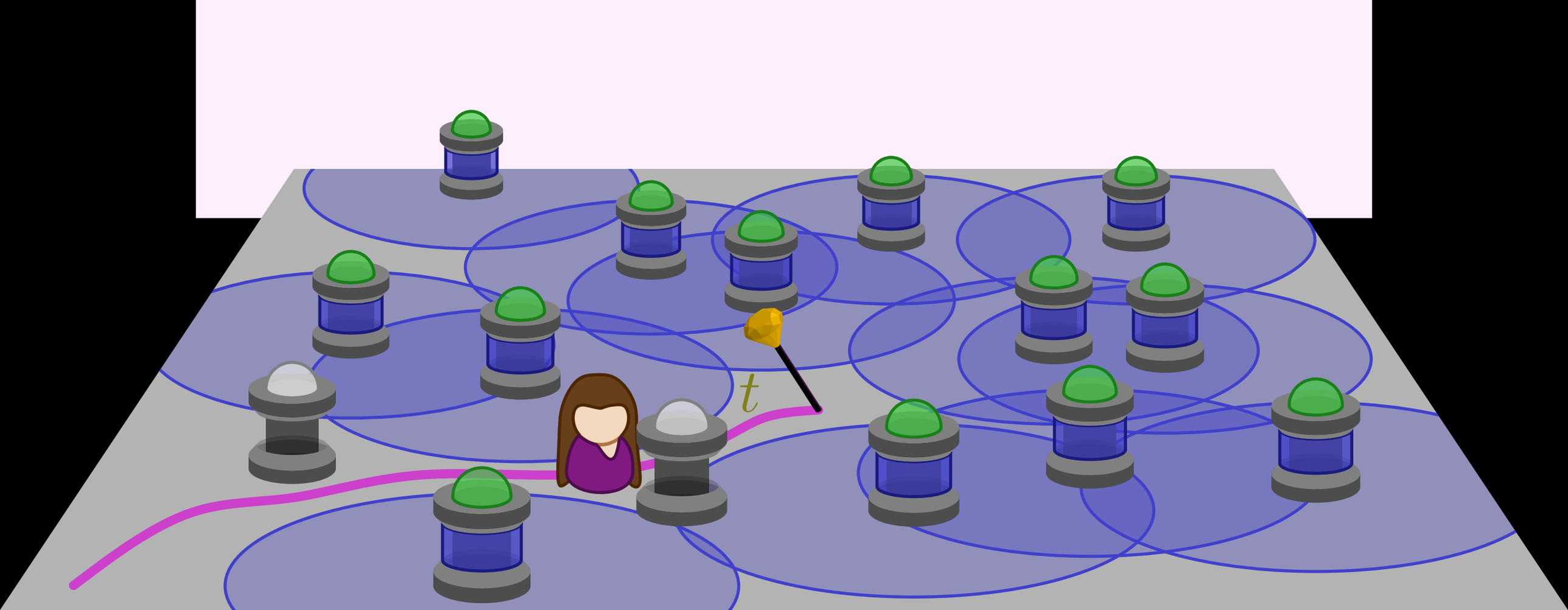
The *resilience* of $\mathcal{R}$ is the smallest number of guards that need to fail before $\mathcal{R}$ no longer guards t.



Sometimes, guards fail.

When they do, t may no longer be guarded!

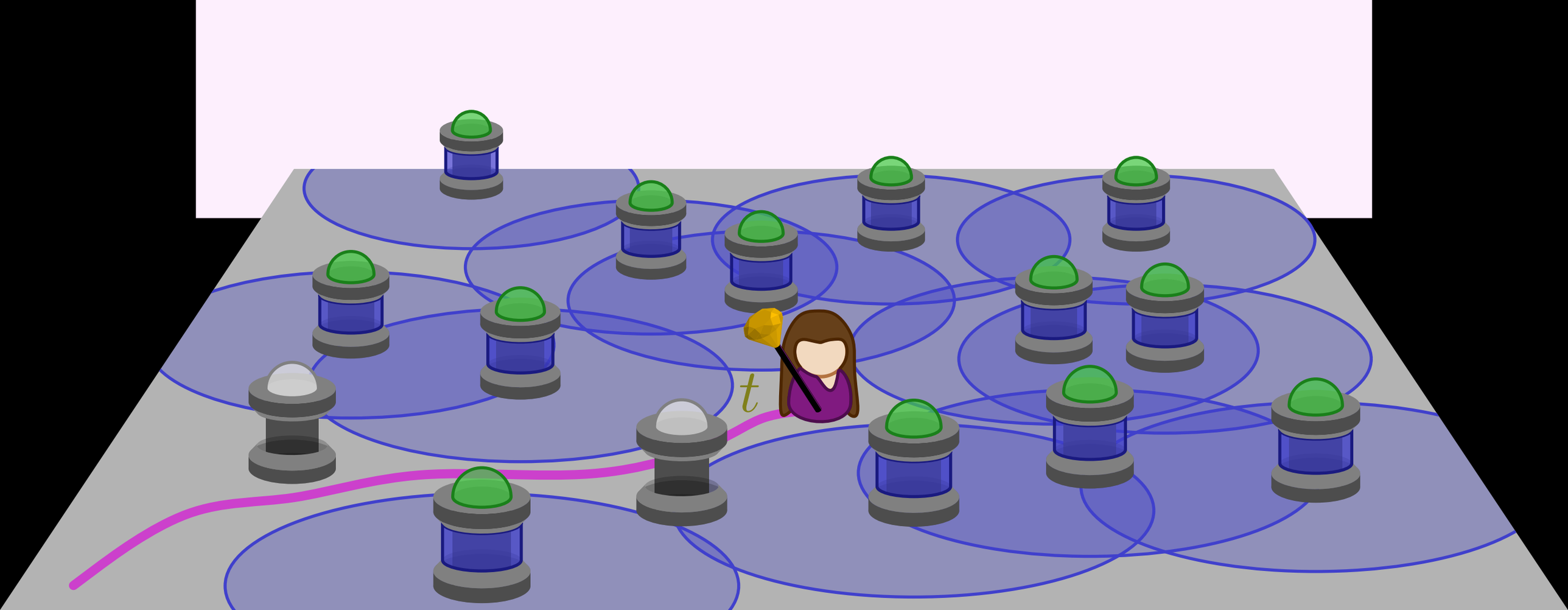
The *resilience* of $\mathcal{R}$ is the smallest number of guards that need to fail before $\mathcal{R}$ no longer guards t.



Sometimes, guards fail.

When they do, t may no longer be guarded!

The *resilience* of $\mathcal{R}$ is the smallest number of guards that need to fail before $\mathcal{R}$ no longer guards t.

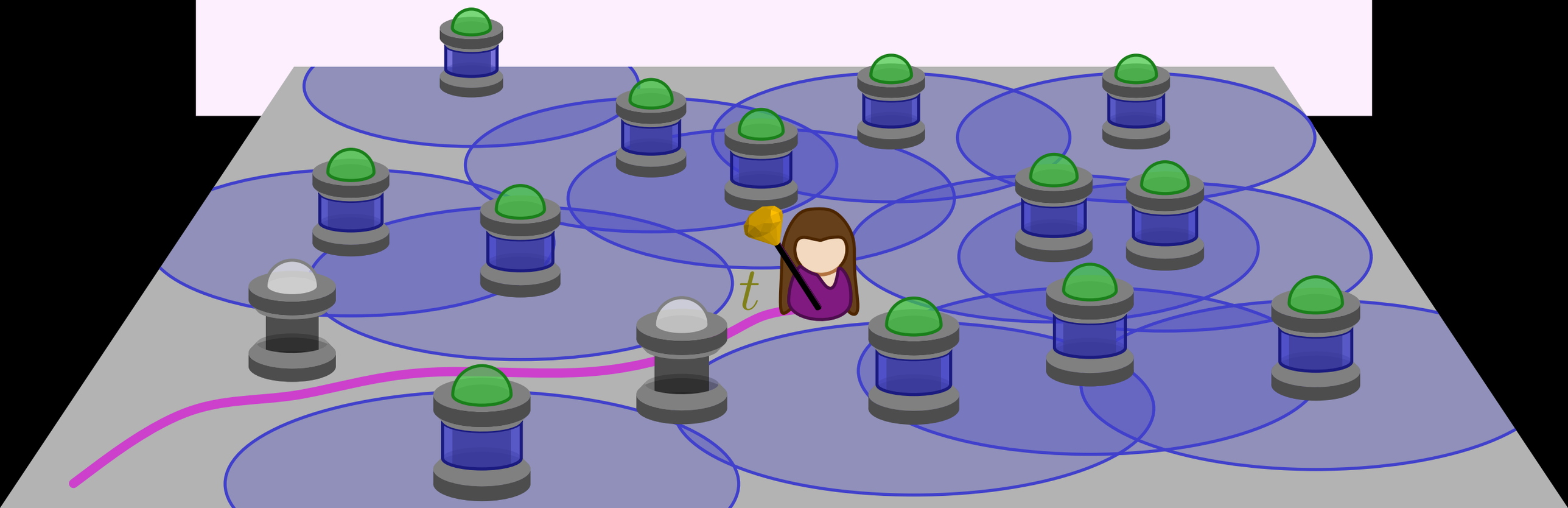


Sometimes, guards fail.

When they do, t may no longer be guarded!

The *resilience* of $\mathcal{R}$ is the smallest number of guards that need to fail before $\mathcal{R}$ no longer guards t.

Can we compute the resilience?



KNOWN & NEW RESULTS

When guarding a *strip*, the resilience can be computed in polynomial time.

(Kumar, Lai & Arora, 2005)

When guarding a *strip*, the resilience can be computed in polynomial time.

(Kumar, Lai & Arora, 2005)

The resilience of unit disks in the plane can be $\frac{5}{6}$-approximated.

(Bereg & Kirkpatrick, 2009)

When guarding a *strip*, the resilience can be computed in polynomial time.

(Kumar, Lai & Arora, 2005)

The resilience of unit disks in the plane can be $\frac{5}{6}$-approximated.

(Bereg & Kirkpatrick, 2009)

Computing the resilience of line segments in the plane is NP-hard.

(Cabello, Giannopoulos & Knauer, 2011)

(Tseng & Kirkpatrick, 2011)

We can compute the resilience r of n unit disks in $O(2^{f(r)}n^3)$ time.

We can compute the resilience r of n unit disks in $O(2^{f(r)}n^3)$ time.

We can ε-approximate the resilience of unit disks of ply δ in $O(2^{f(\delta,\varepsilon)}n^7)$ time.

We can compute the resilience r of n unit disks in $O(2^{f(r)}n^3)$ time.

We can ε-approximate the resilience of unit disks of ply δ in $O(2^{f(\delta,\varepsilon)}n^7)$ time.

These results extend to β-fat regions.

We can compute the resilience r of n unit disks in $O(2^{f(r)}n^3)$ time.

We can ε-approximate the resilience of unit disks of ply δ in $O(2^{f(\delta,\varepsilon)}n^7)$ time.

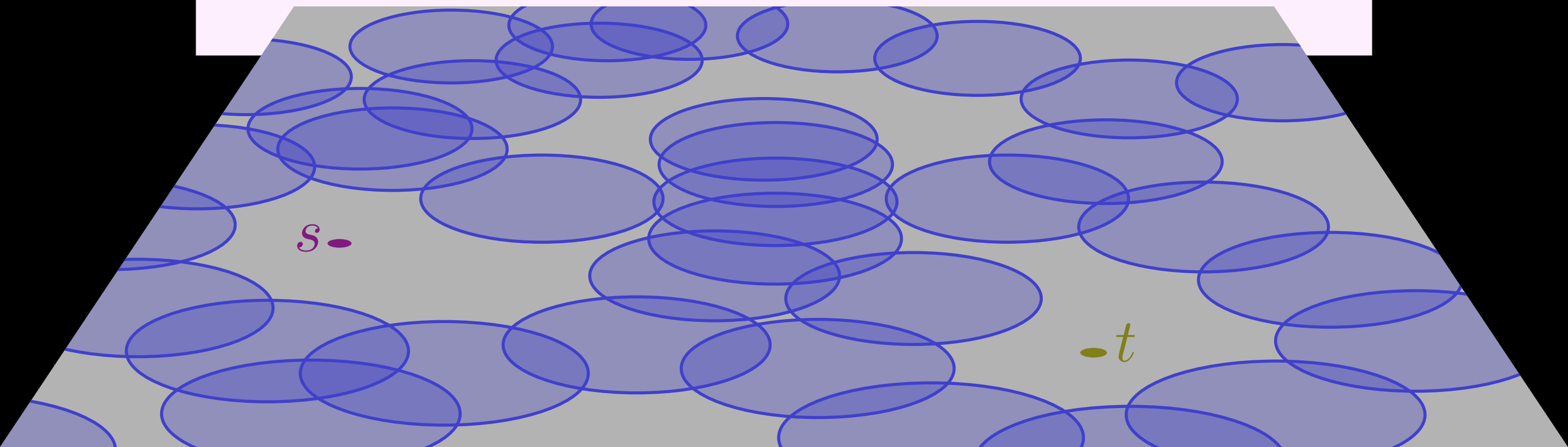
These results extend to β-fat regions.

Computing the resilience of β-fat regions is NP-hard.

FIXED PARAMETER ALGORITHM

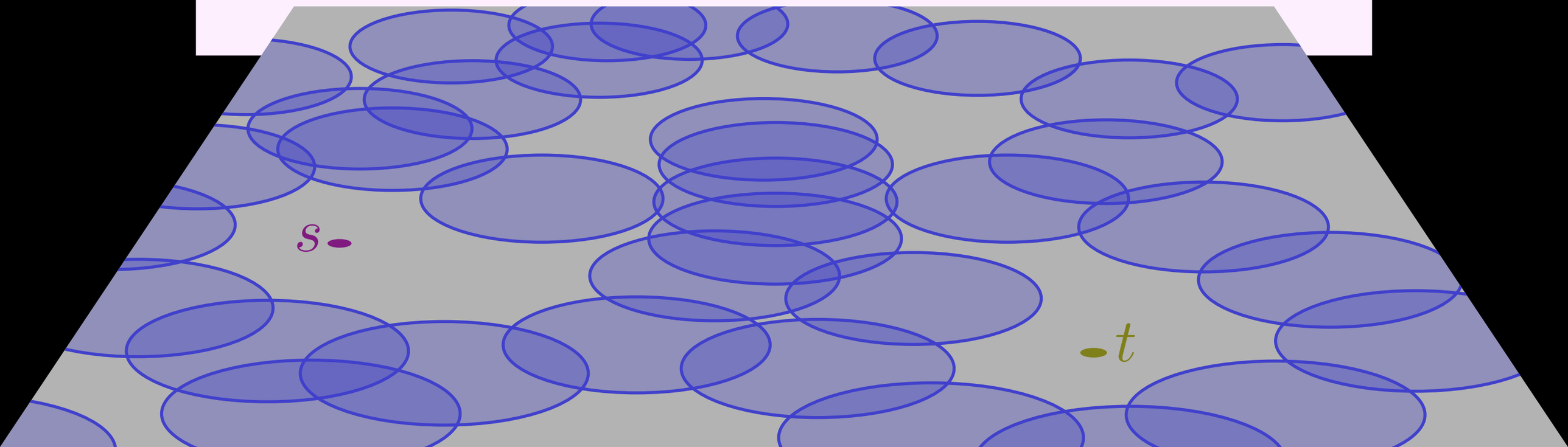
Consider two points s and t and a set of unit disks $\mathcal{R}$.

Consider two points s and t and a set of unit disks $\mathcal{R}$.



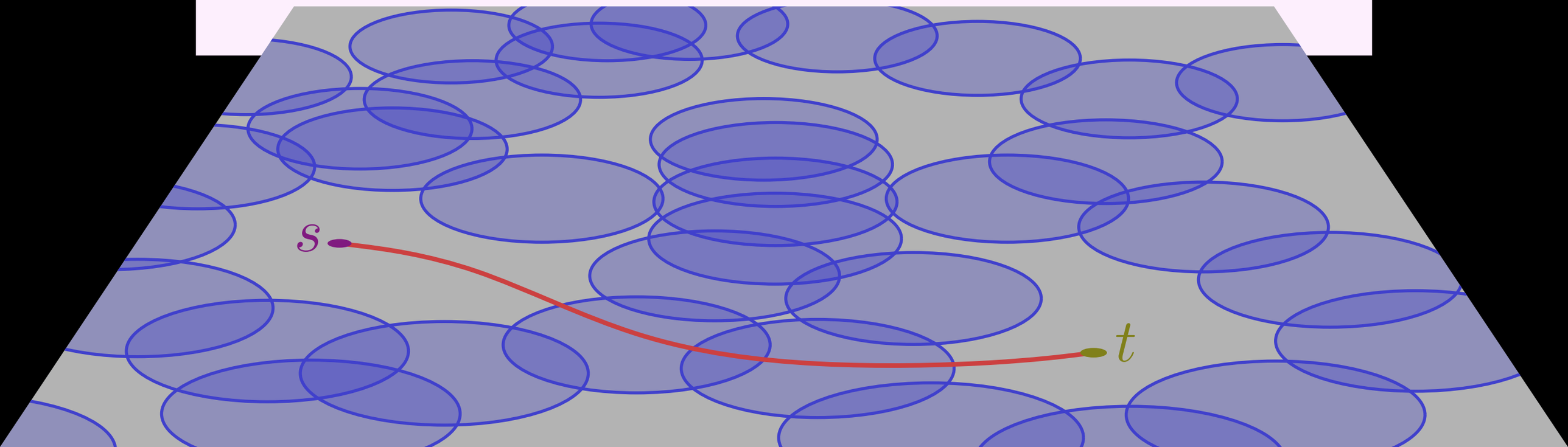
Consider two points s and t and a set of unit disks $\mathcal{R}$.

The *thickness* or $\mathcal{R}$ is half the shortest path from s to t in the arrangement.



Consider two points s and t and a set of unit disks $\mathcal{R}$.

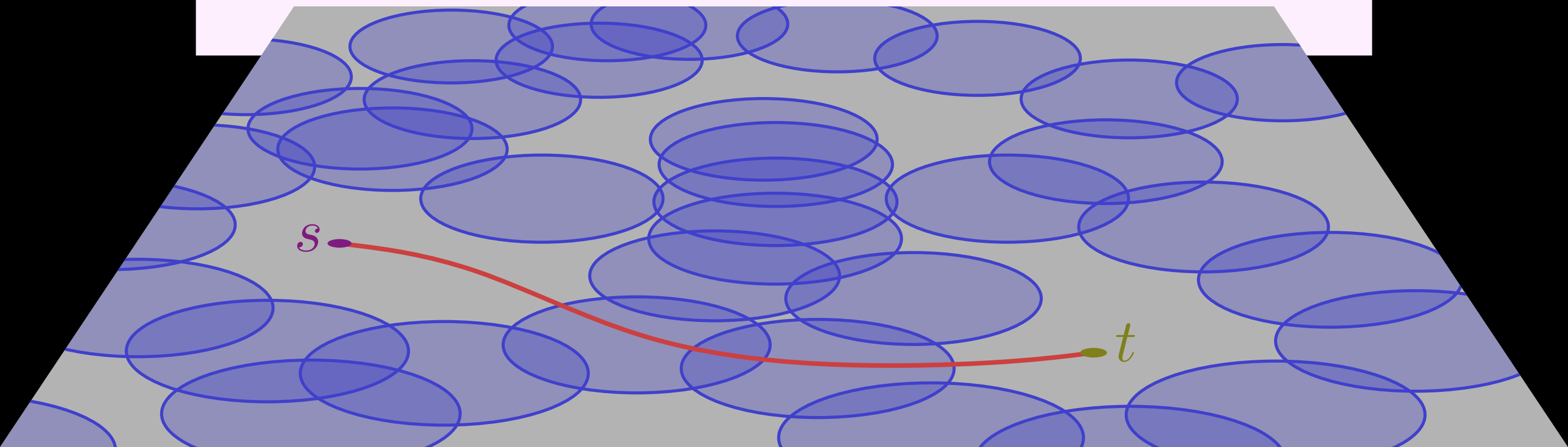
The *thickness* or $\mathcal{R}$ is half the shortest path from s to t in the arrangement.



Consider two points s and t and a set of unit disks $\mathcal{R}$.

The *thickness* or $\mathcal{R}$ is half the shortest path from s to t in the arrangement.

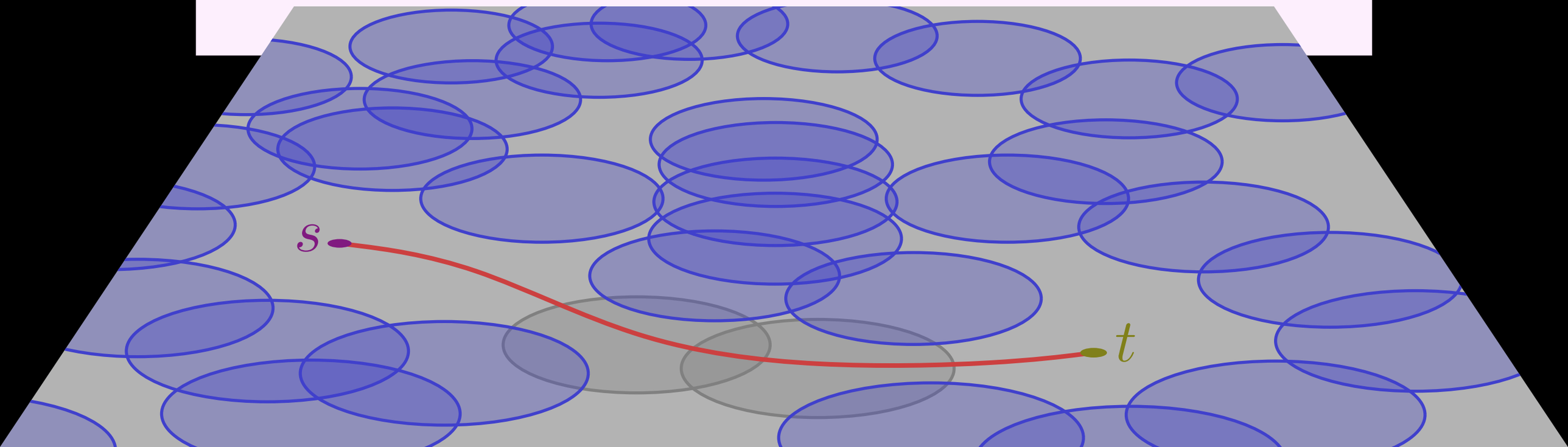
This path gives an upper bound on the resilience.



Consider two points s and t and a set of unit disks $\mathcal{R}$.

The *thickness* or $\mathcal{R}$ is half the shortest path from s to t in the arrangement.

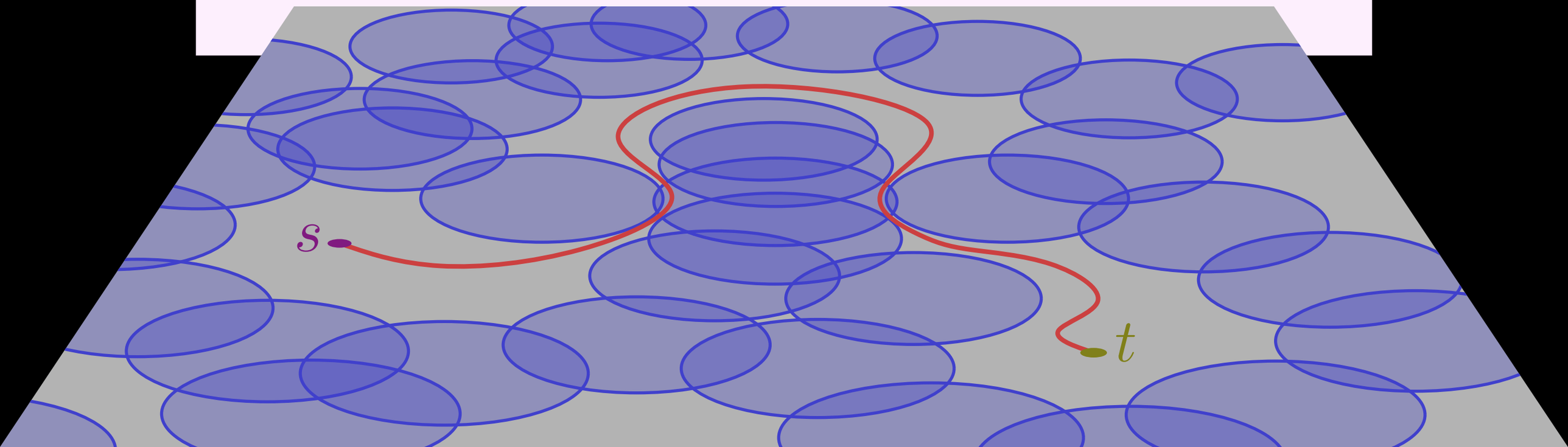
This path gives an upper bound on the resilience.



Consider two points s and t and a set of unit disks $\mathcal{R}$.

The *thickness* or $\mathcal{R}$ is half the shortest path from s to t in the arrangement.

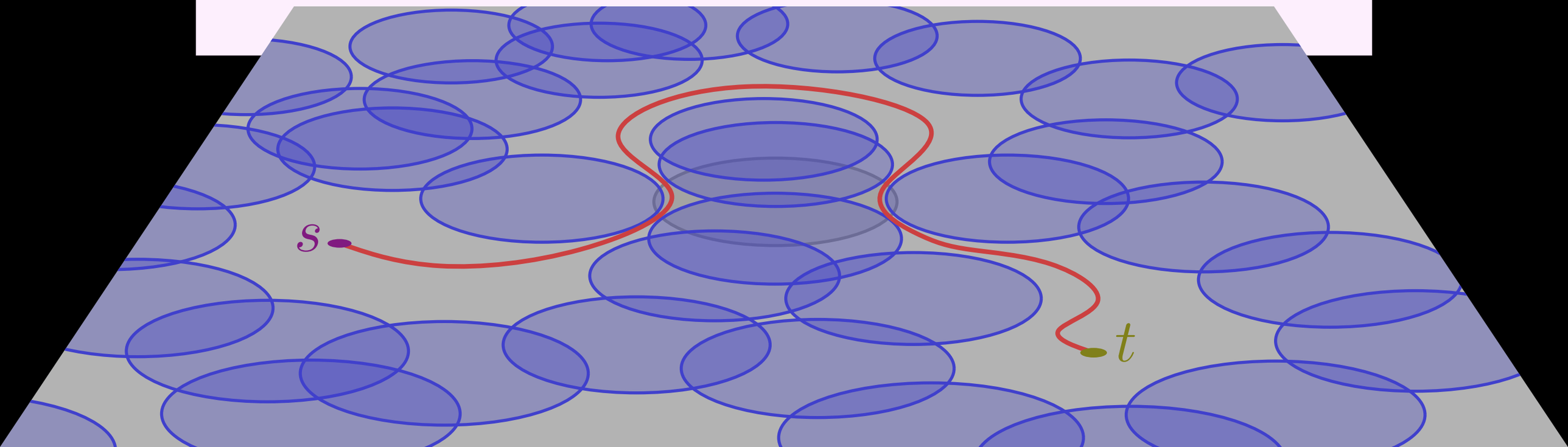
This path gives an upper bound on the resilience.



Consider two points s and t and a set of unit disks $\mathcal{R}$.

The *thickness* or $\mathcal{R}$ is half the shortest path from s to t in the arrangement.

This path gives an upper bound on the resilience.

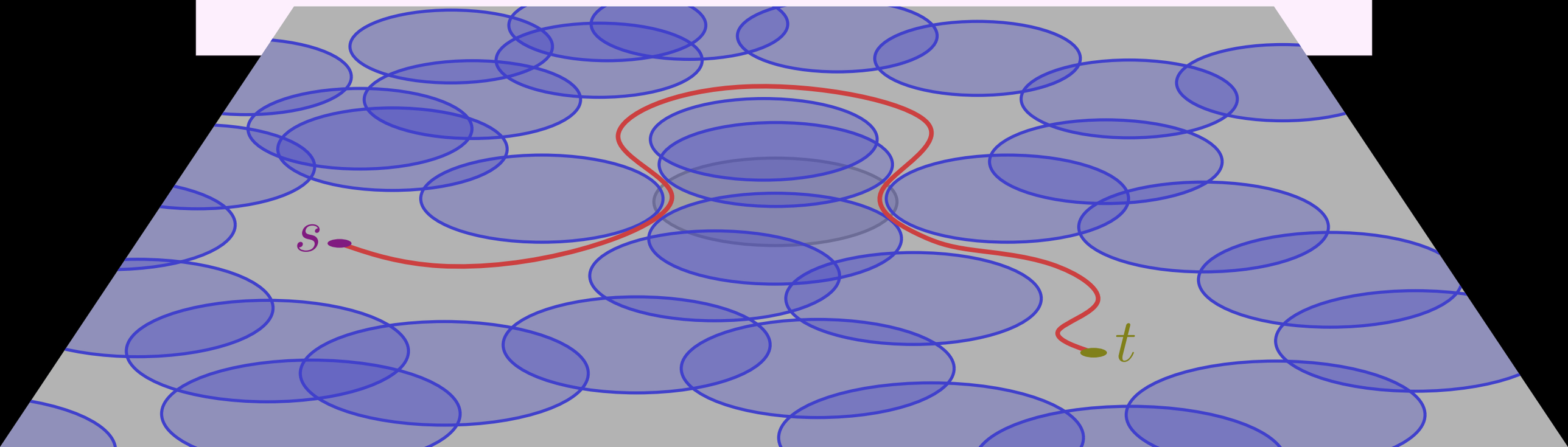


Consider two points s and t and a set of unit disks $\mathcal{R}$.

The *thickness* or $\mathcal{R}$ is half the shortest path from s to t in the arrangement.

This path gives an upper bound on the resilience.

LEMMA: The thickness is at most twice the resilience. (Bereg & Kirkpatrick, 2009)

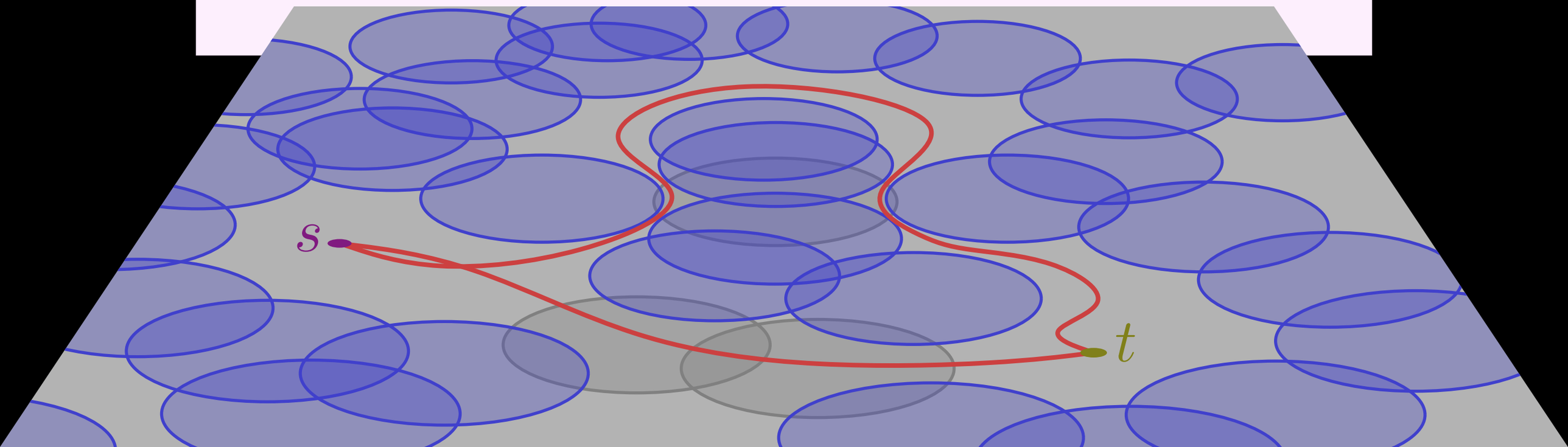


Consider two points s and t and a set of unit disks $\mathcal{R}$.

The *thickness* or $\mathcal{R}$ is half the shortest path from s to t in the arrangement.

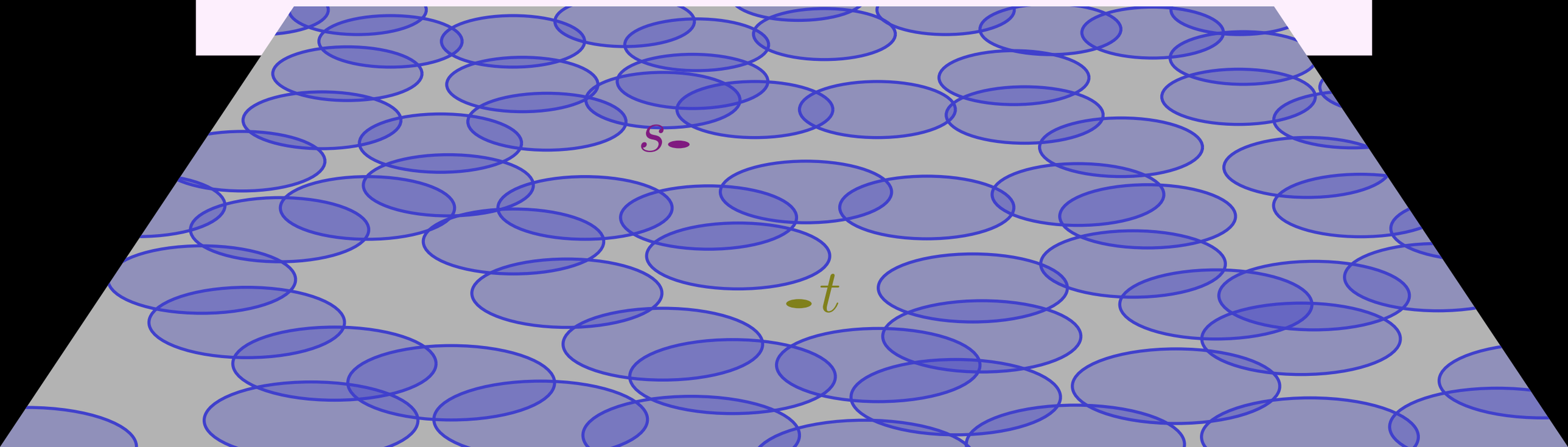
This path gives an upper bound on the resilience.

LEMMA: The thickness is at most twice the resilience. (Bereg & Kirkpatrick, 2009)



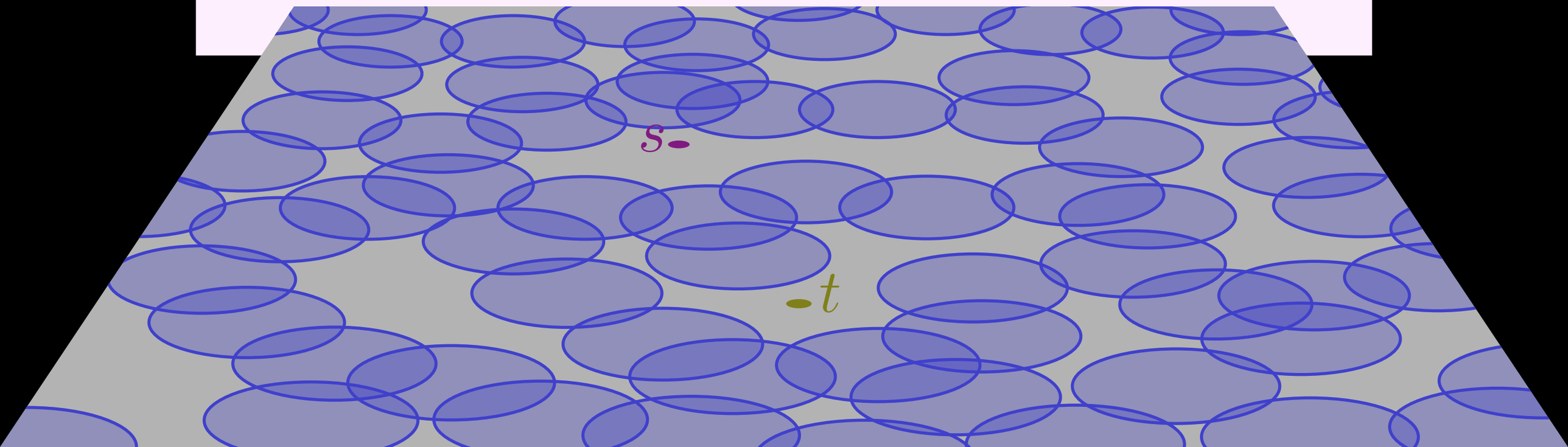
Let r be our estimated resilience.

Let r be our estimated resilience.



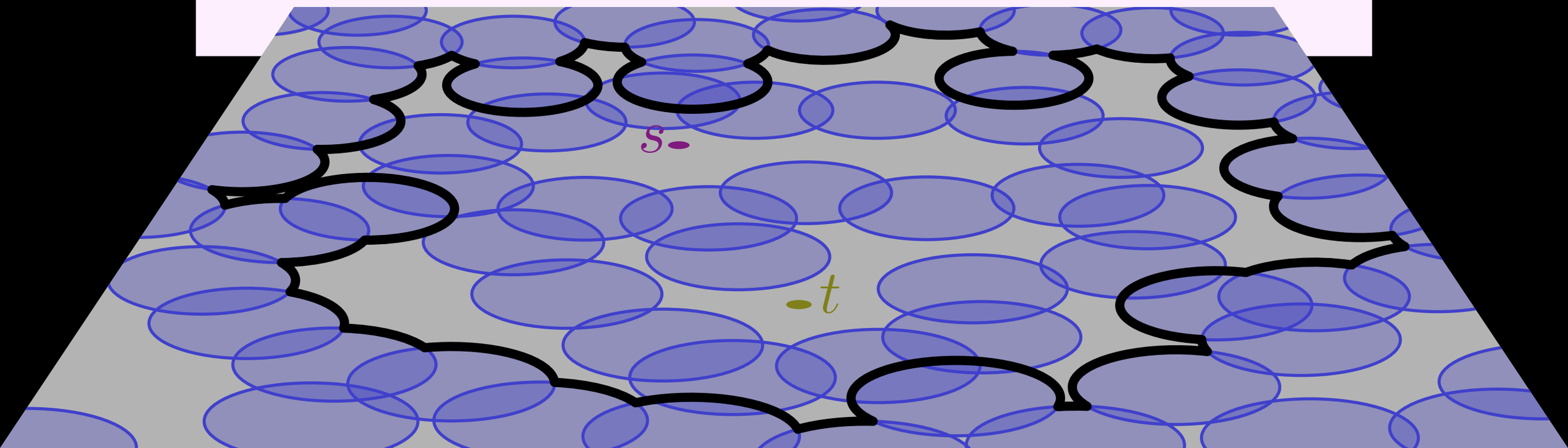
Let r be our estimated resilience.

We only need to consider the part of $\mathcal{R}$ within distance $O(r)$ from s and t.



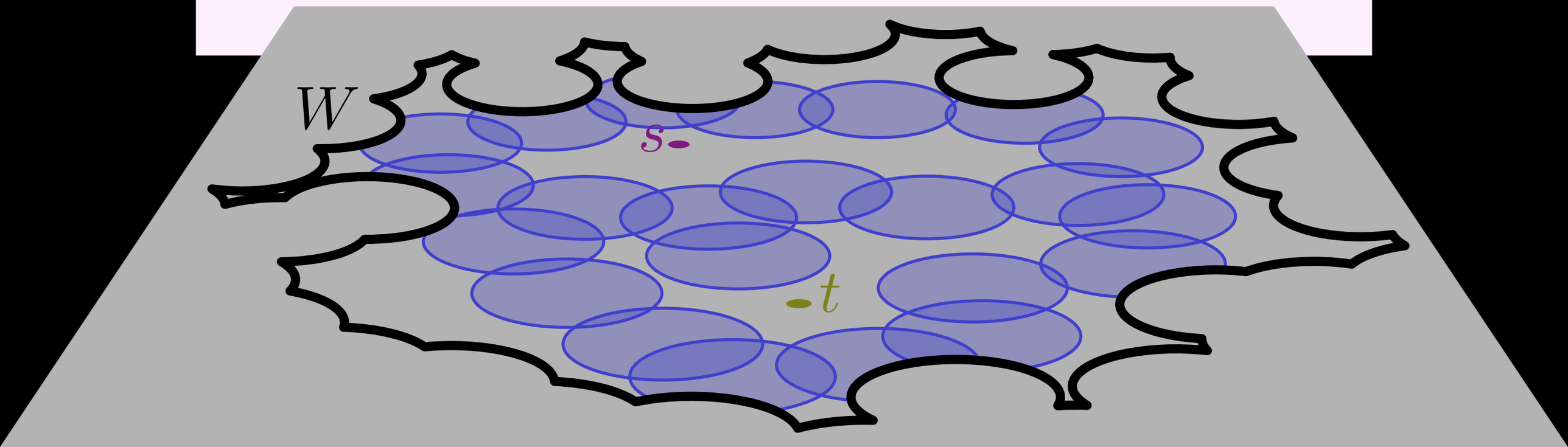
Let r be our estimated resilience.

We only need to consider the part of $\mathcal{R}$ within distance $O(r)$ from s and t.



Let r be our estimated resilience.

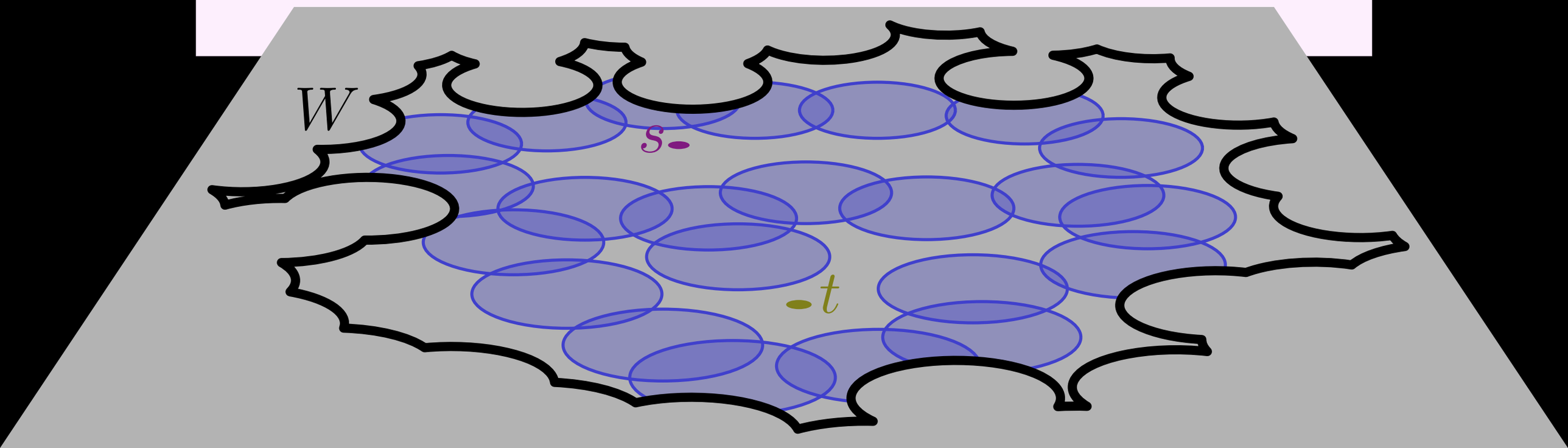
We only need to consider the part of $\mathcal{R}$ within distance $O(r)$ from s and t.



Let r be our estimated resilience.

We only need to consider the part of $\mathcal{R}$ within distance $O(r)$ from s and t.

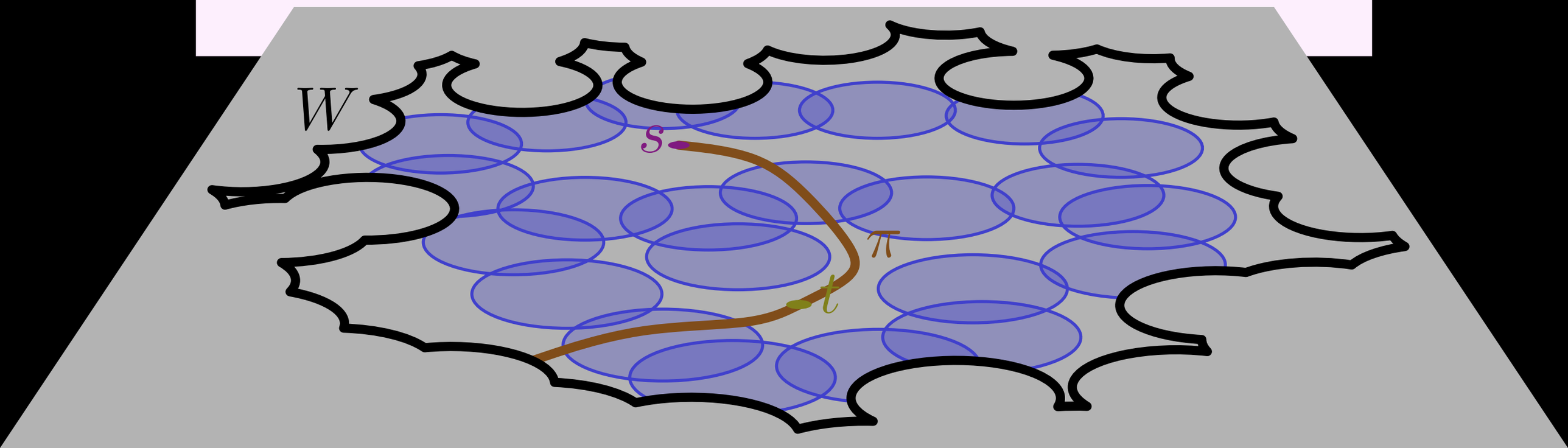
LEMMA: There exists a path π from s via t to the boundary of W that goes through at most $O(r)$ disks.



Let r be our estimated resilience.

We only need to consider the part of $\mathcal{R}$ within distance $O(r)$ from s and t.

LEMMA: There exists a path π from s via t to the boundary of W that goes through at most $O(r)$ disks.

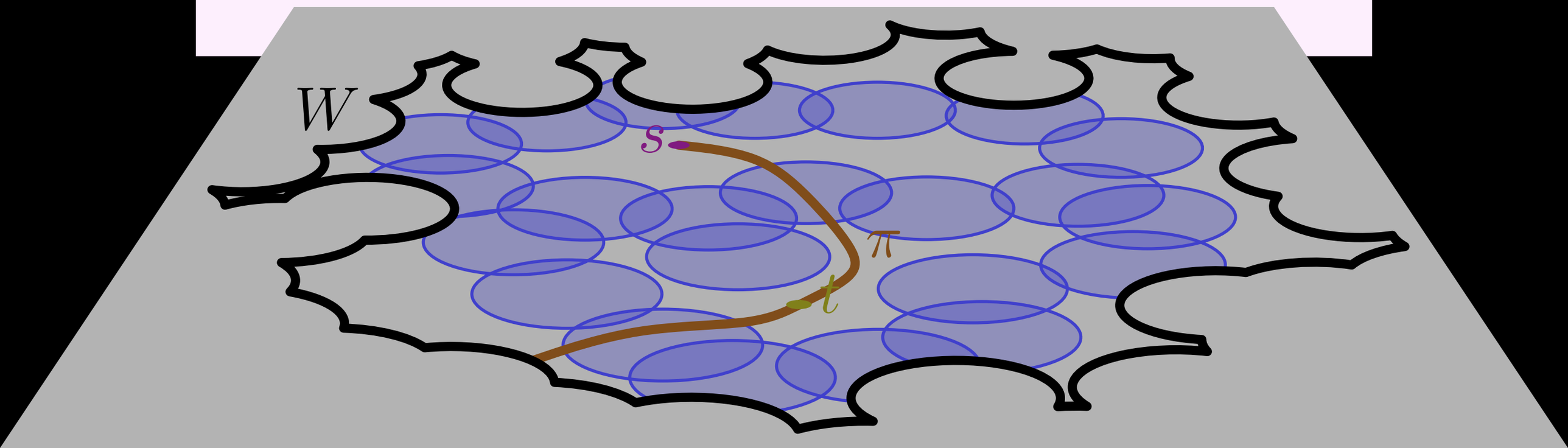


Let r be our estimated resilience.

We only need to consider the part of $\mathcal{R}$ within distance $O(r)$ from s and t.

LEMMA: There exists a path π from s via t to the boundary of W that goes through at most $O(r)$ disks.

We guess which of these are in the solution: only $2^{O(r)}$ possibilities.

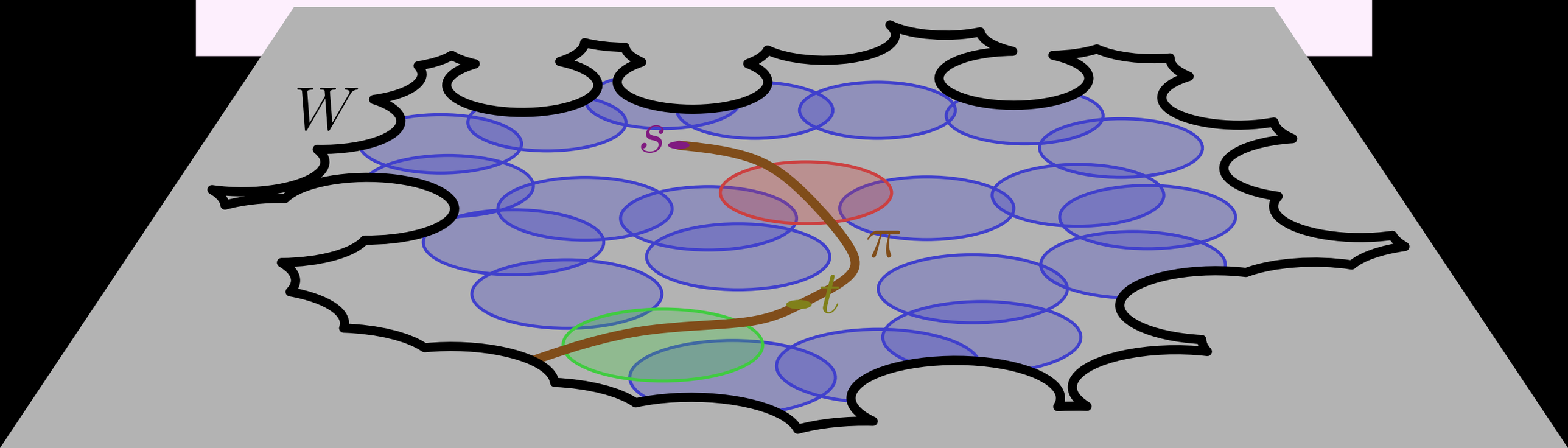


Let r be our estimated resilience.

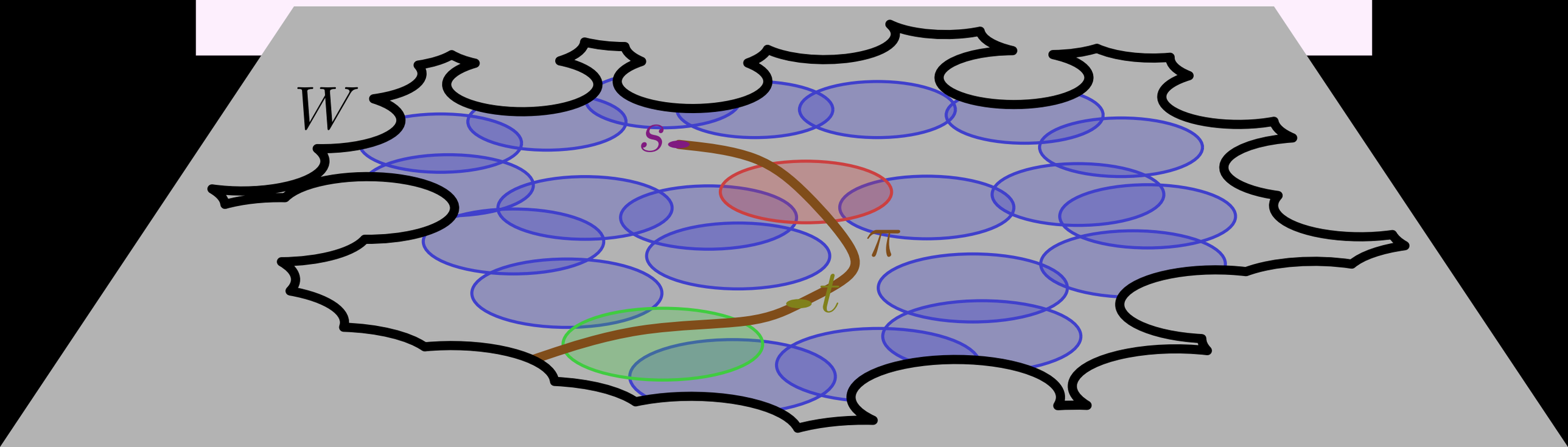
We only need to consider the part of $\mathcal{R}$ within distance $O(r)$ from s and t.

LEMMA: There exists a path π from s via t to the boundary of W that goes through at most $O(r)$ disks.

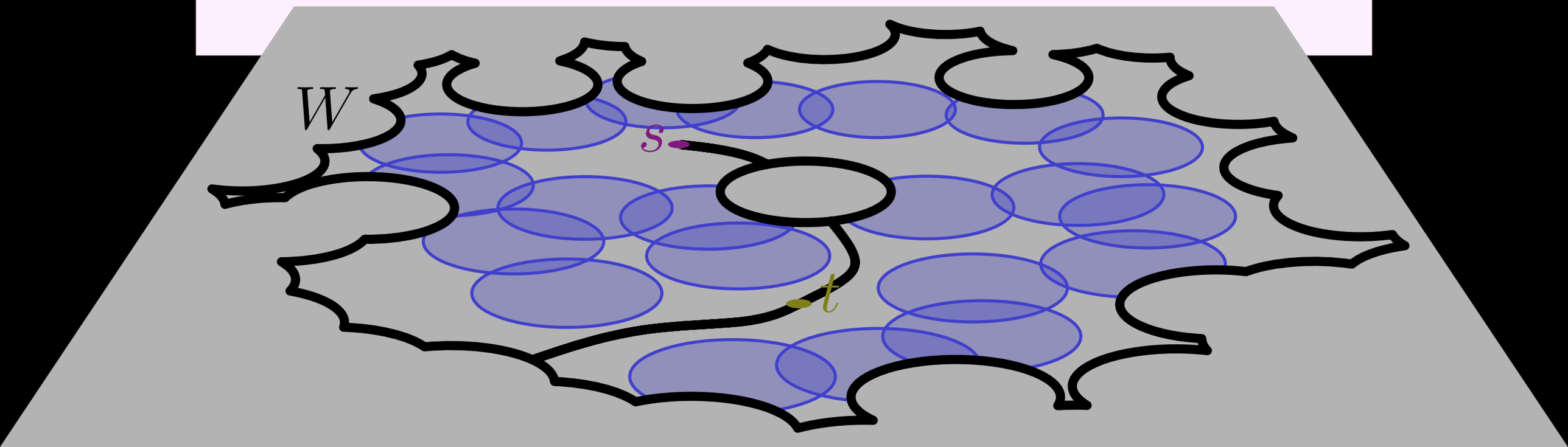
We guess which of these are in the solution: only $2^{O(r)}$ possibilities.



IDEA: Cut open the domain along π.

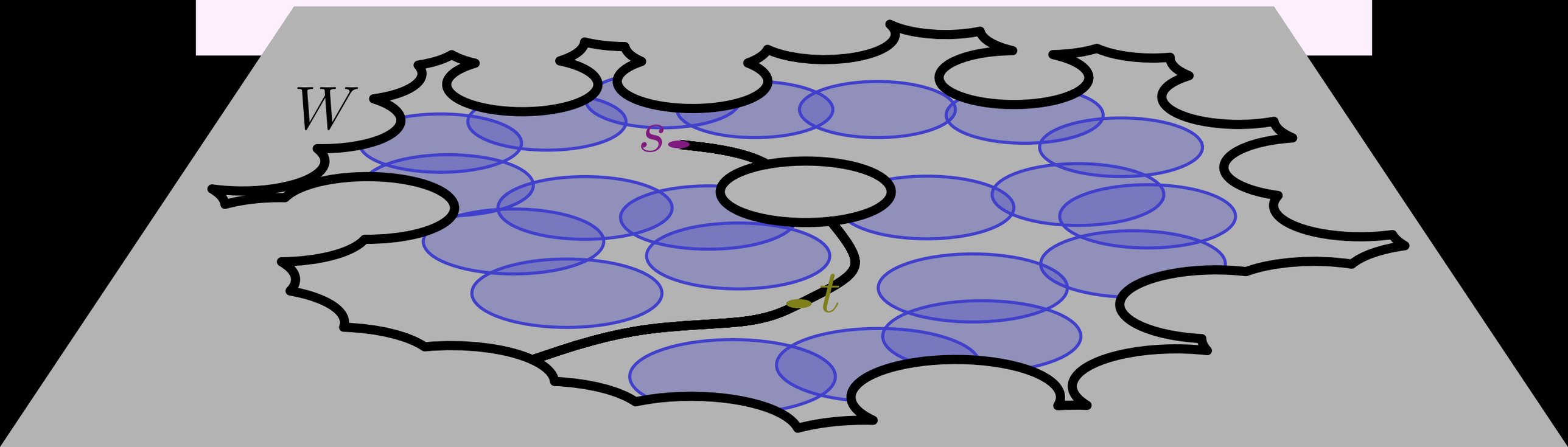


IDEA: Cut open the domain along π.



IDEA: Cut open the domain along π.

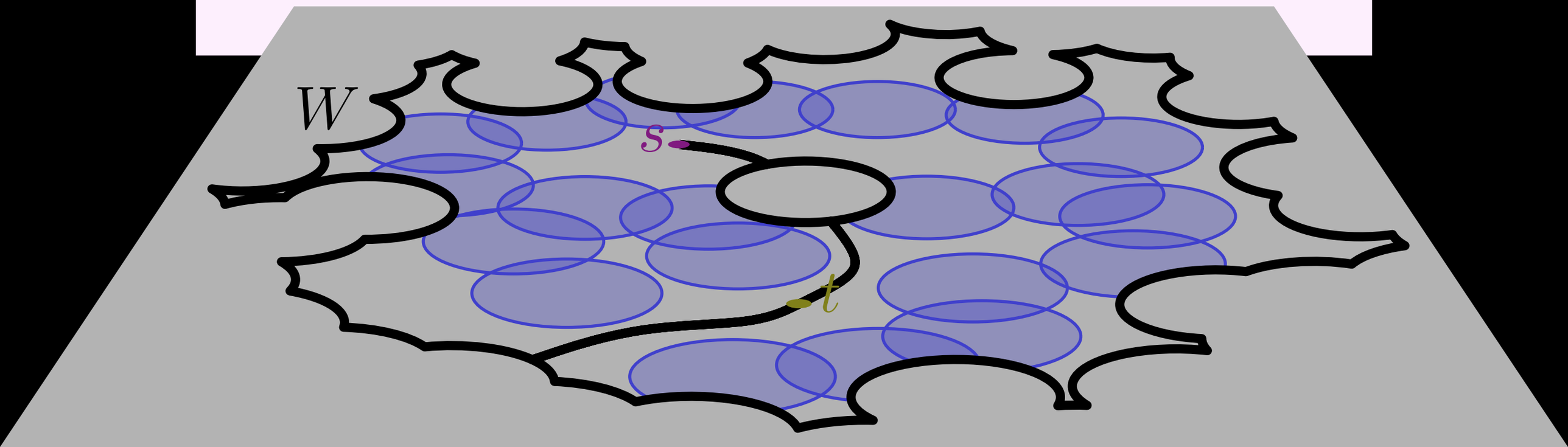
Now the problem looks *almost* like computing the resilience of a strip!



IDEA: Cut open the domain along π.

Now the problem looks *almost* like computing the resilience of a strip!

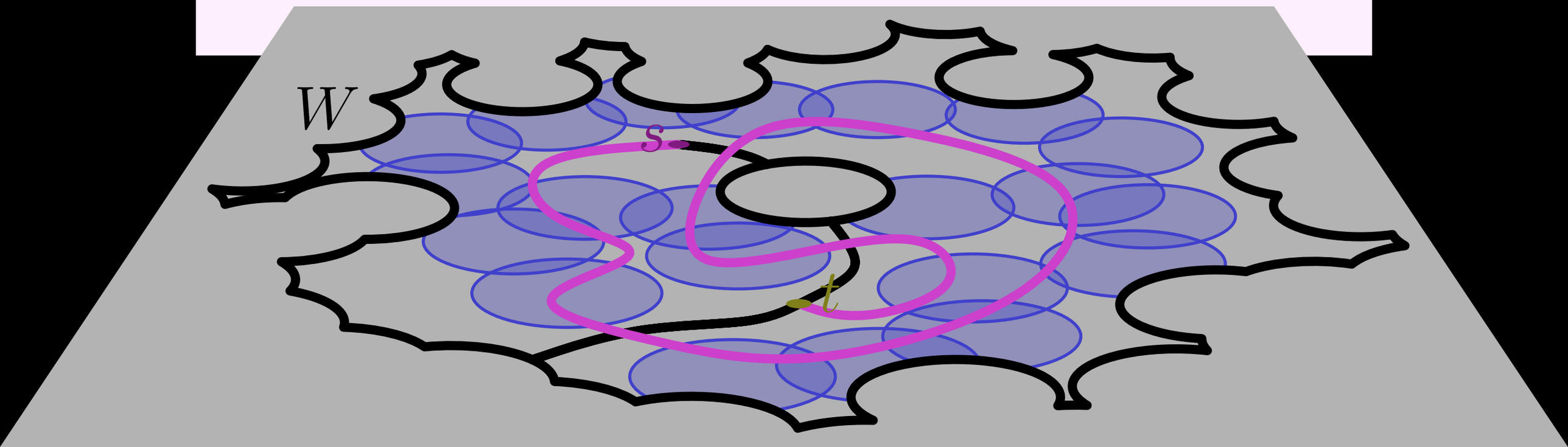
. . . but, the optimal solution may still cross π . . .



IDEA: Cut open the domain along π.

Now the problem looks *almost* like computing the resilience of a strip!

. . . but, the optimal solution may still cross π . . .

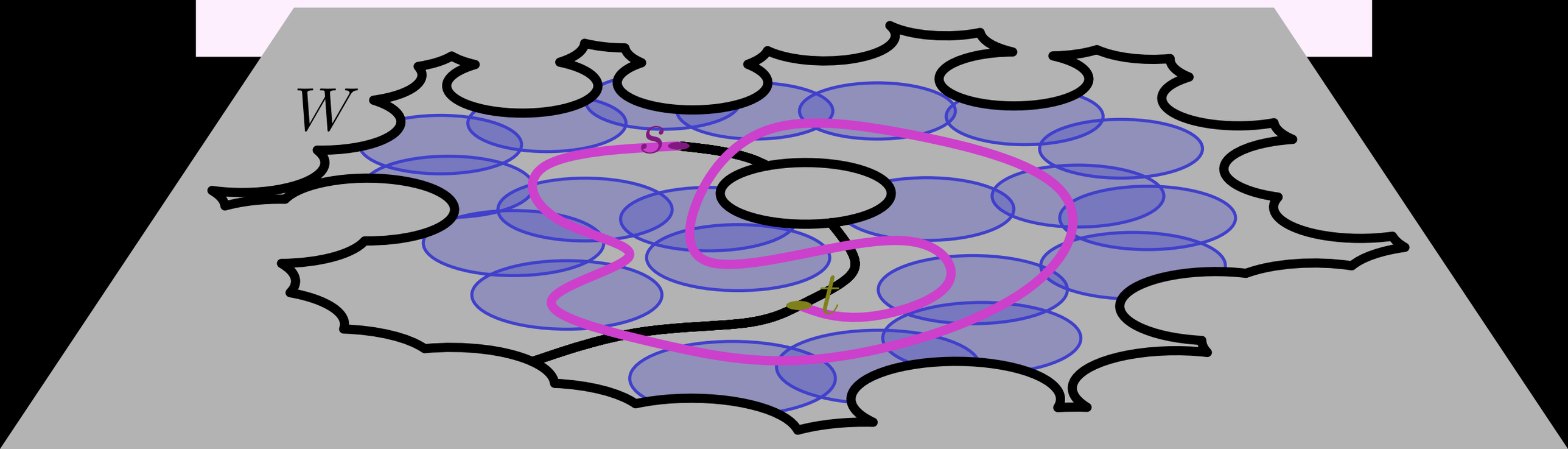


IDEA: Cut open the domain along π.

Now the problem looks *almost* like computing the resilience of a strip!

. . . but, the optimal solution may still cross π . . .

LEMMA: There are only $2^{O(r)}$ possible crossings patterns.



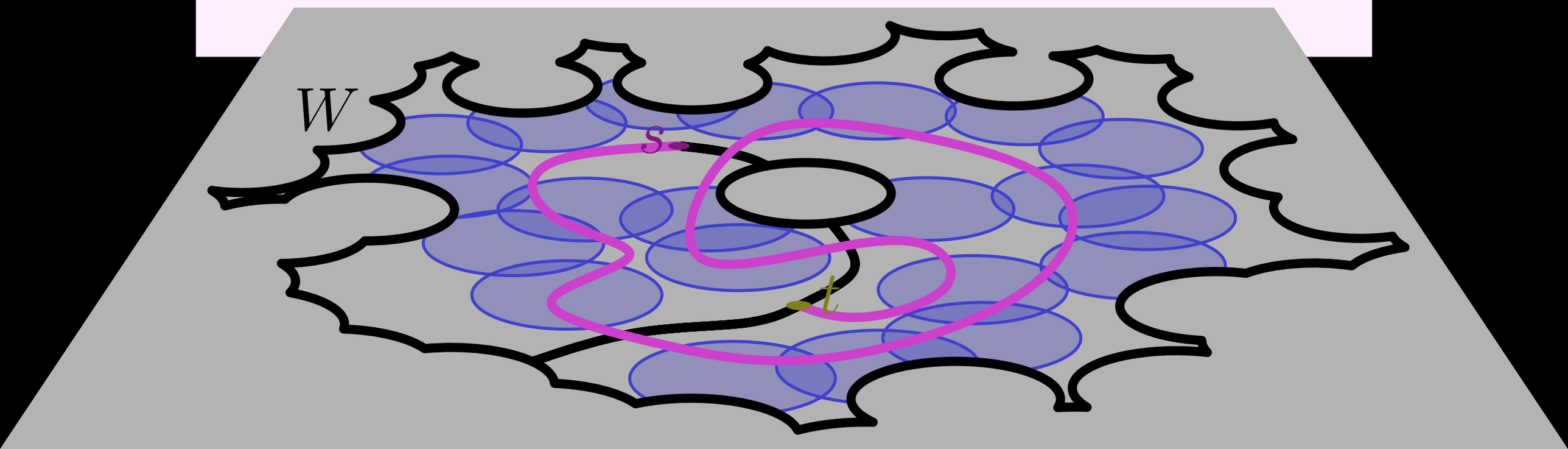
IDEA: Cut open the domain along π.

Now the problem looks *almost* like computing the resilience of a strip!

. . . but, the optimal solution may still cross π . . .

LEMMA: There are only $2^{O(r)}$ possible crossings patterns.

So. . . we just try them all.



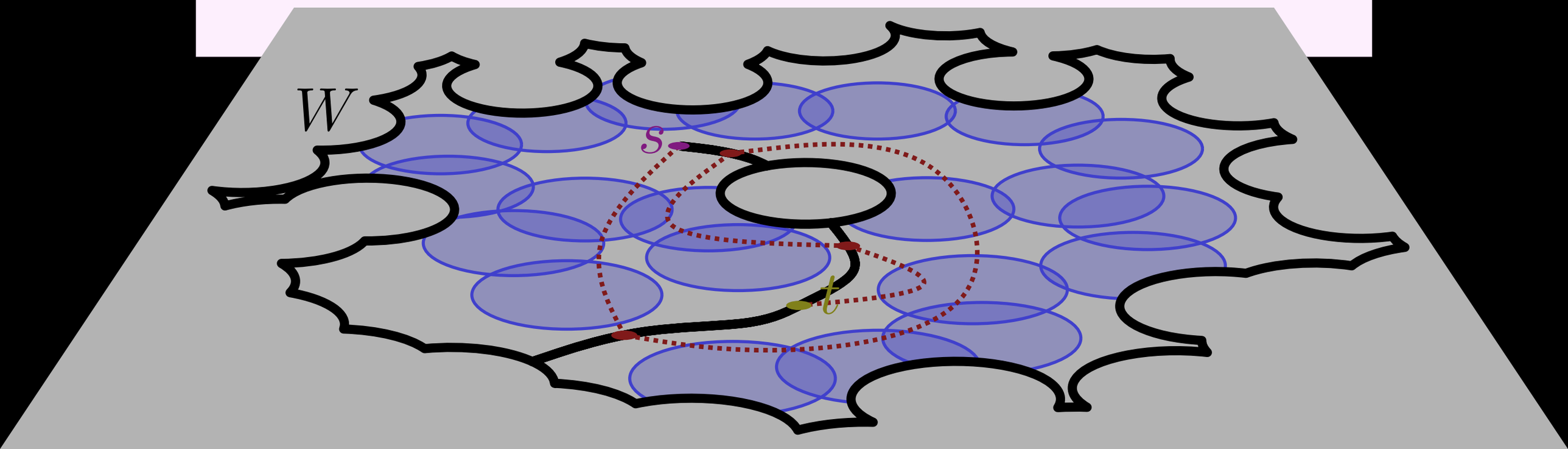
IDEA: Cut open the domain along π.

Now the problem looks *almost* like computing the resilience of a strip!

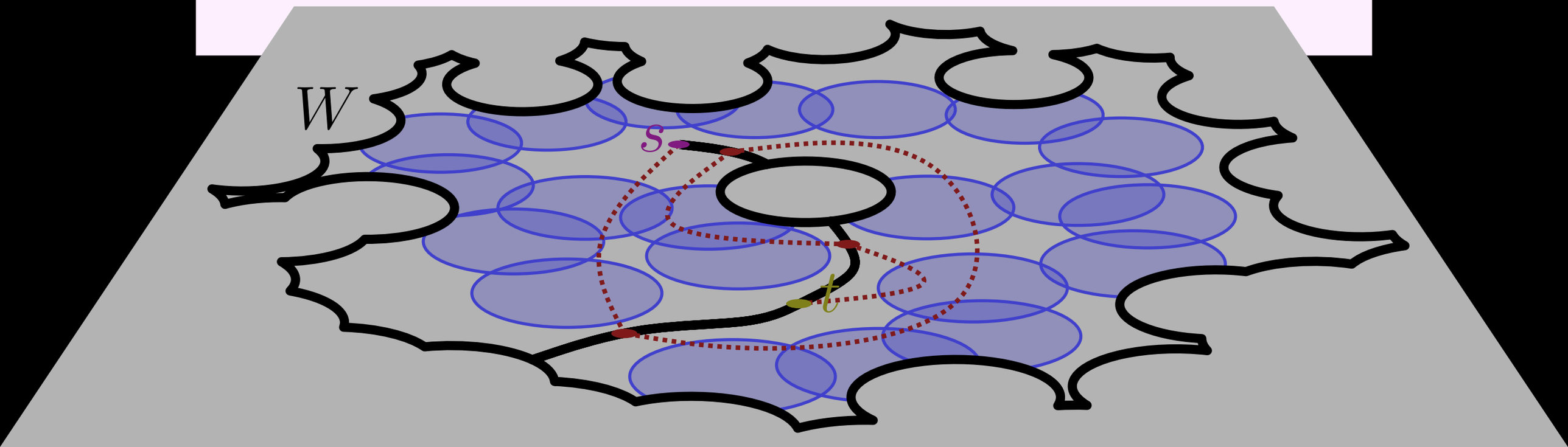
. . . but, the optimal solution may still cross π . . .

LEMMA: There are only $2^{O(r)}$ possible crossings patterns.

So. . . we just try them all.

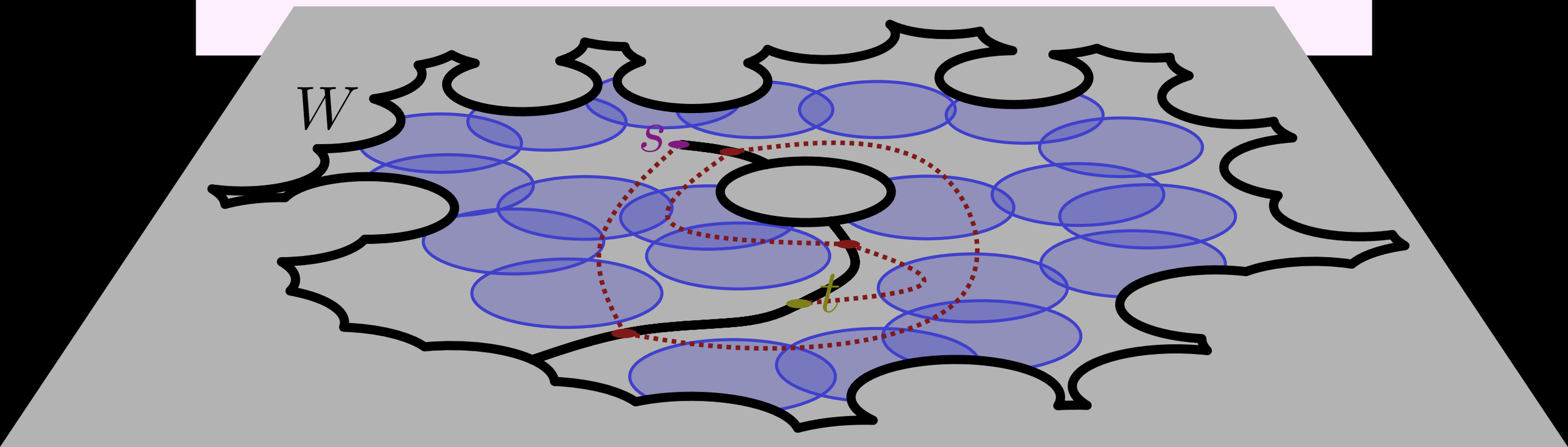


Now we have a nice, clean problem.



Now we have a nice, clean problem.

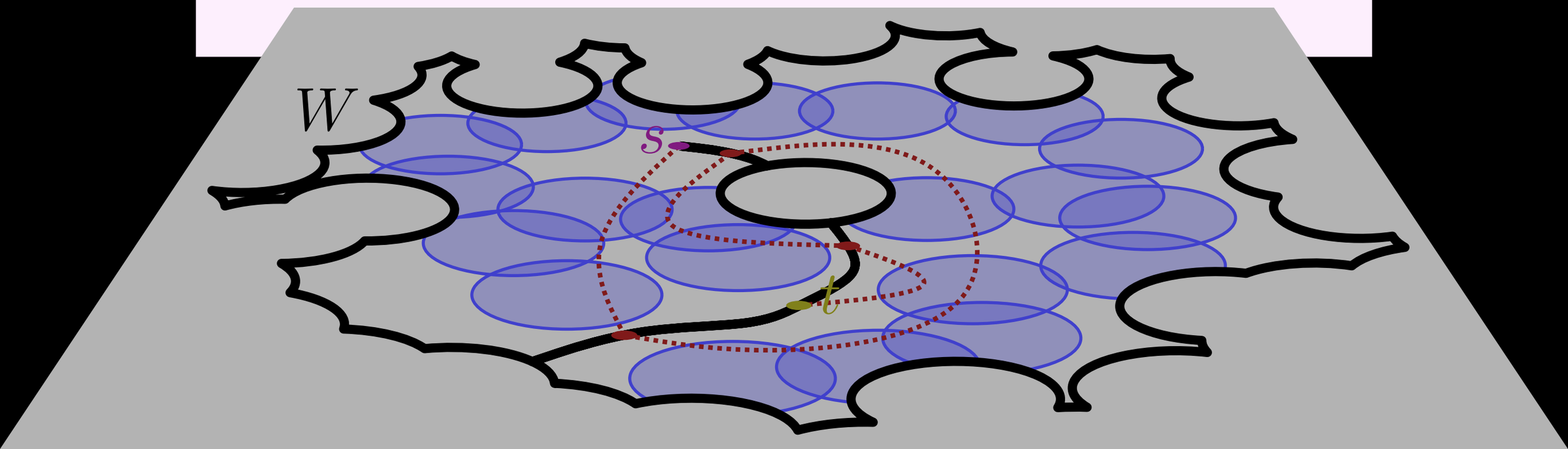
Given a simply connected region with pairs of points on the boundary, remove the smallest number of disks such that all pairs are connected.



Now we have a nice, clean problem.

Given a simply connected region with pairs of points on the boundary, remove the smallest number of disks such that all pairs are connected.

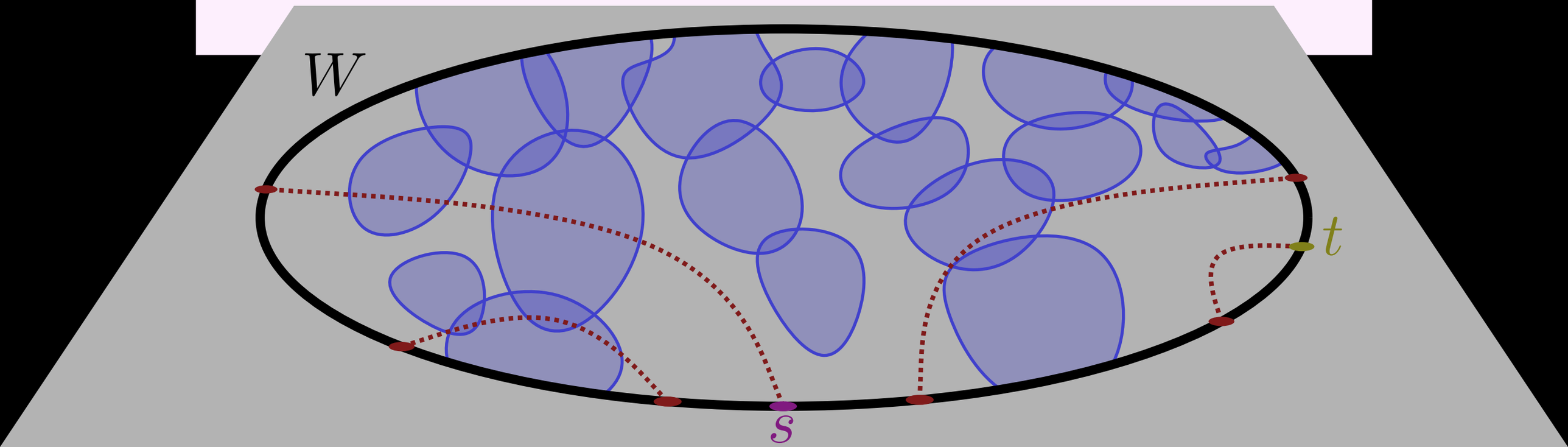
OBSERVATION: The geometry no longer matters.



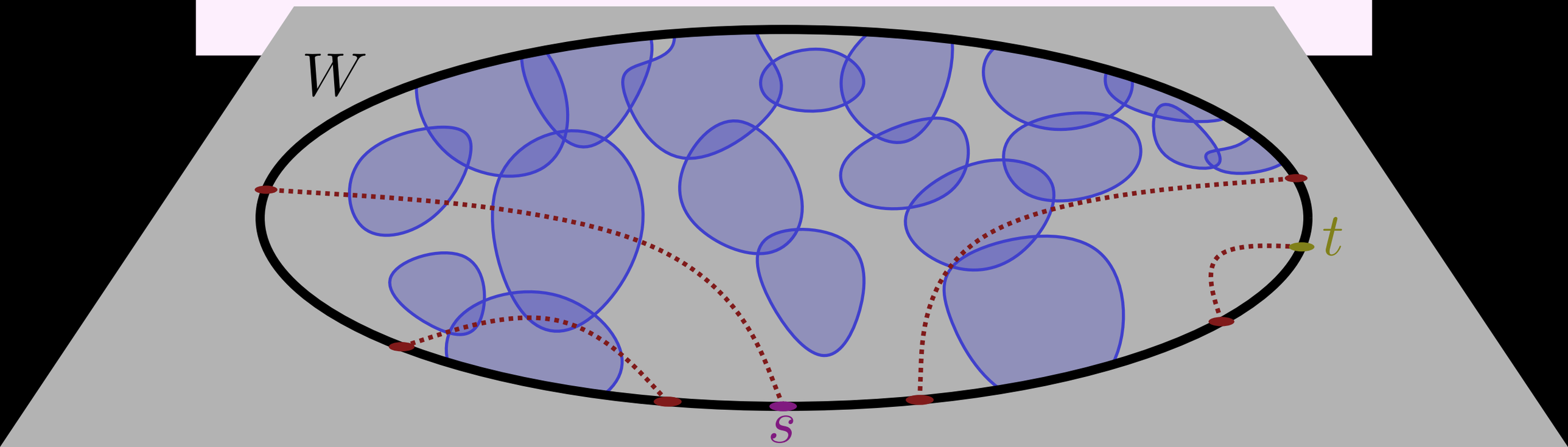
Now we have a nice, clean problem.

Given a simply connected region with pairs of points on the boundary, remove the smallest number of disks such that all pairs are connected.

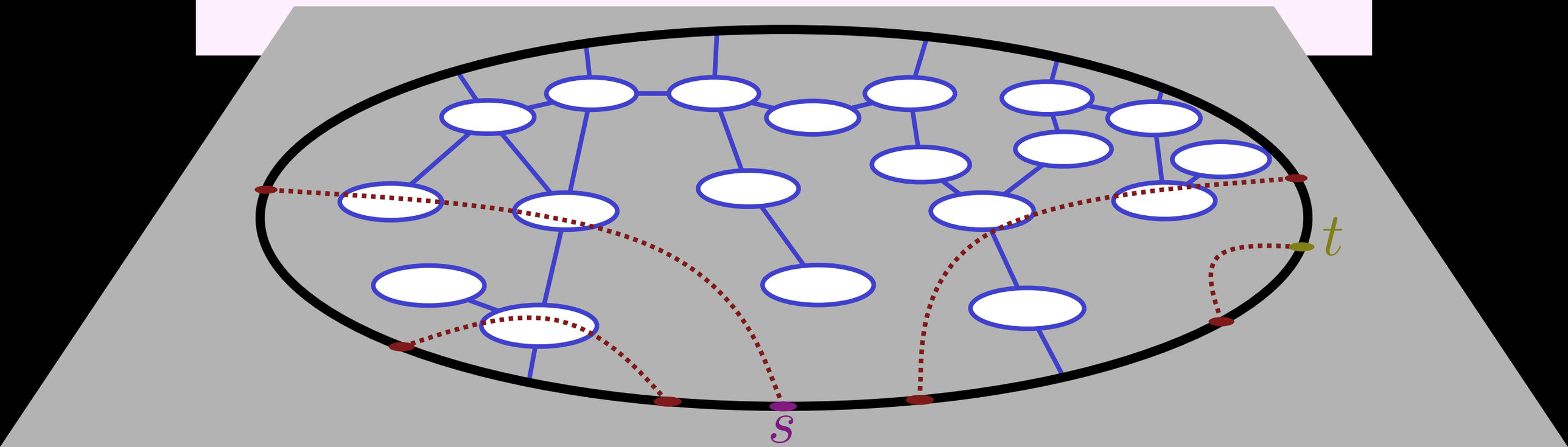
OBSERVATION: The geometry no longer matters.



Consider the intersection graph.

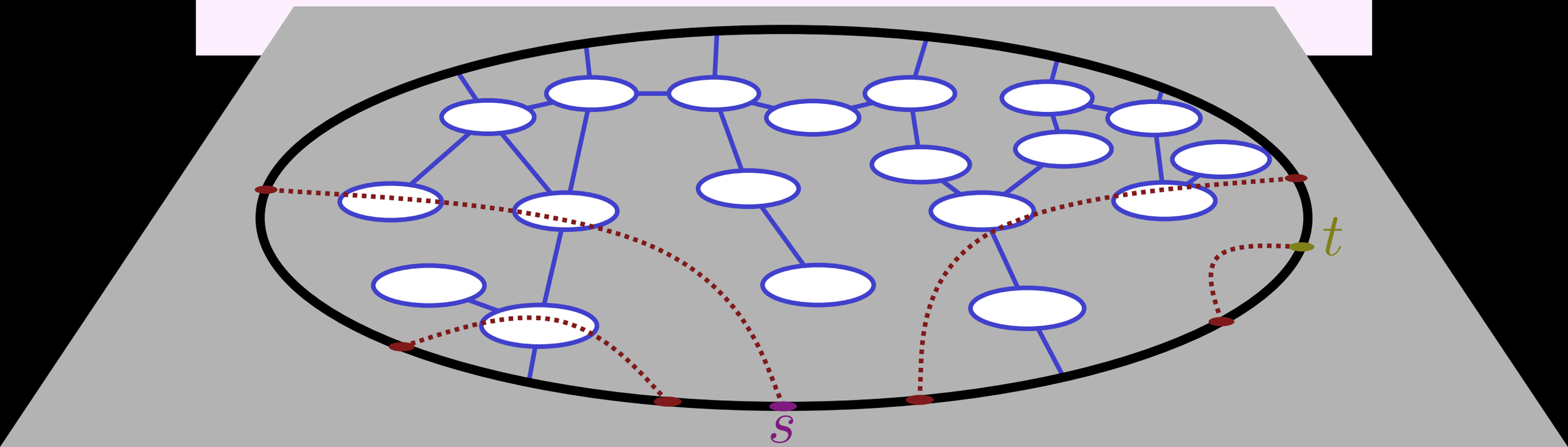


Consider the intersection graph.



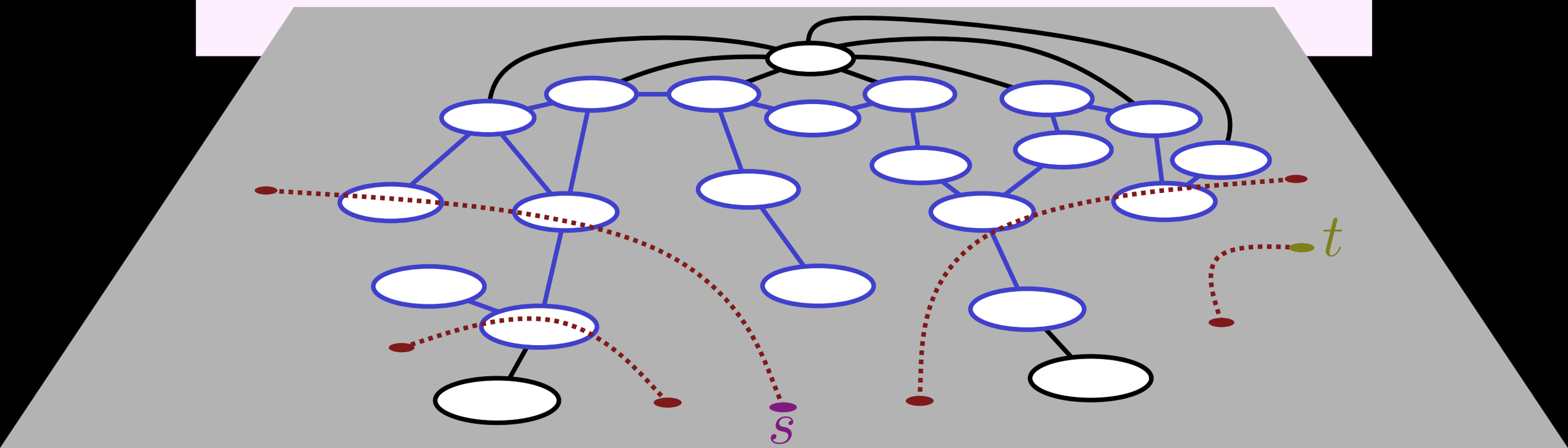
Consider the intersection graph.

Add special *terminal* vertices for pieces of boundary (at most $O(r)$).



Consider the intersection graph.

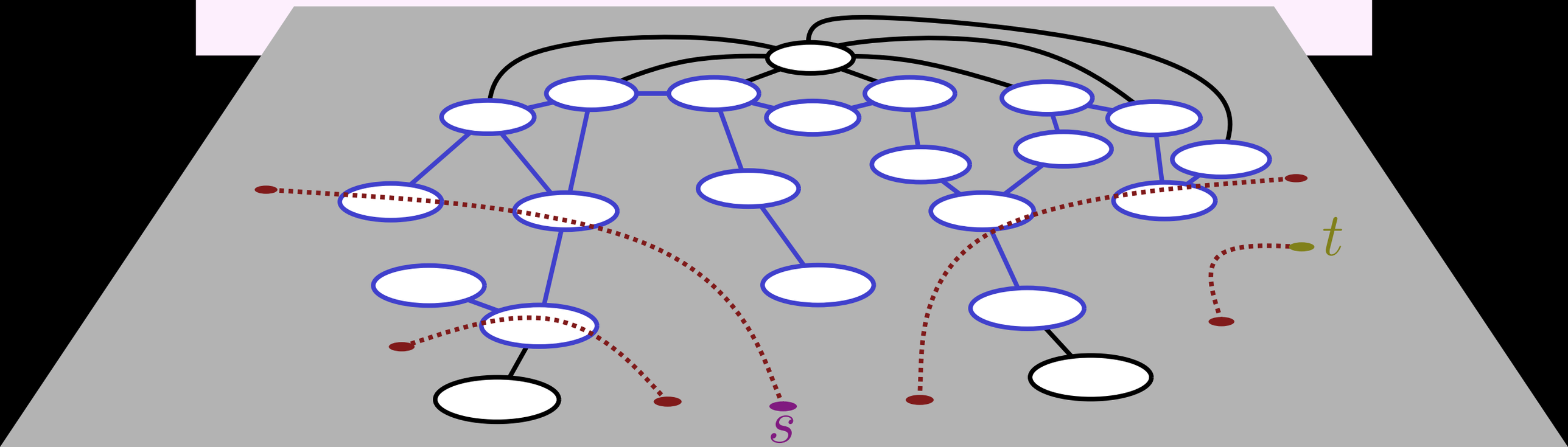
Add special *terminal* vertices for pieces of boundary (at most $O(r)$).



Consider the intersection graph.

Add special *terminal* vertices for pieces of boundary (at most $O(r)$).

We want to *cut* this graph "along the dotted lines".

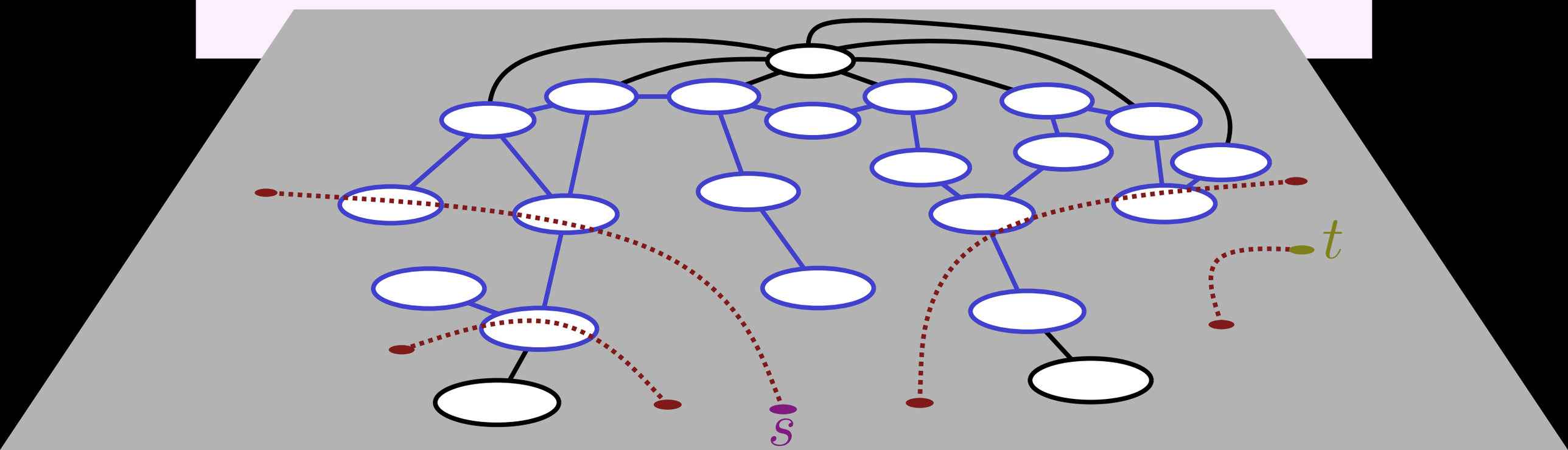


Consider the intersection graph.

Add special *terminal* vertices for pieces of boundary (at most $O(r)$).

We want to *cut* this graph "along the dotted lines".

OBSERVATION: This is the same as cutting between all pairs of terminal vertices that are separated by at least one dotted line.

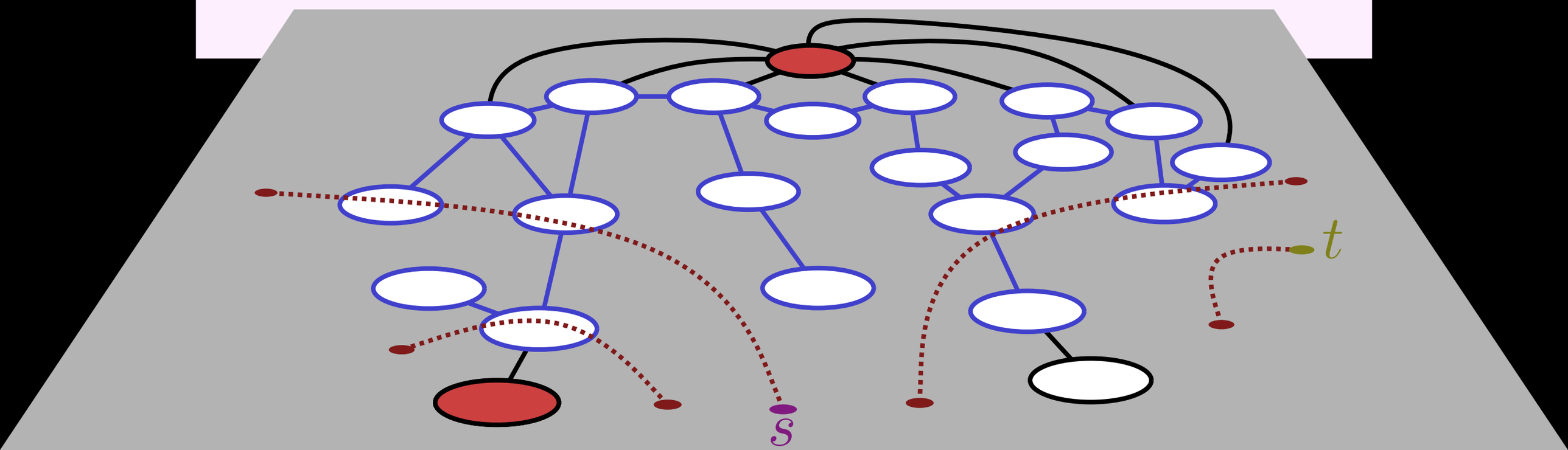


Consider the intersection graph.

Add special *terminal* vertices for pieces of boundary (at most $O(r)$).

We want to *cut* this graph "along the dotted lines".

OBSERVATION: This is the same as cutting between all pairs of terminal vertices that are separated by at least one dotted line.

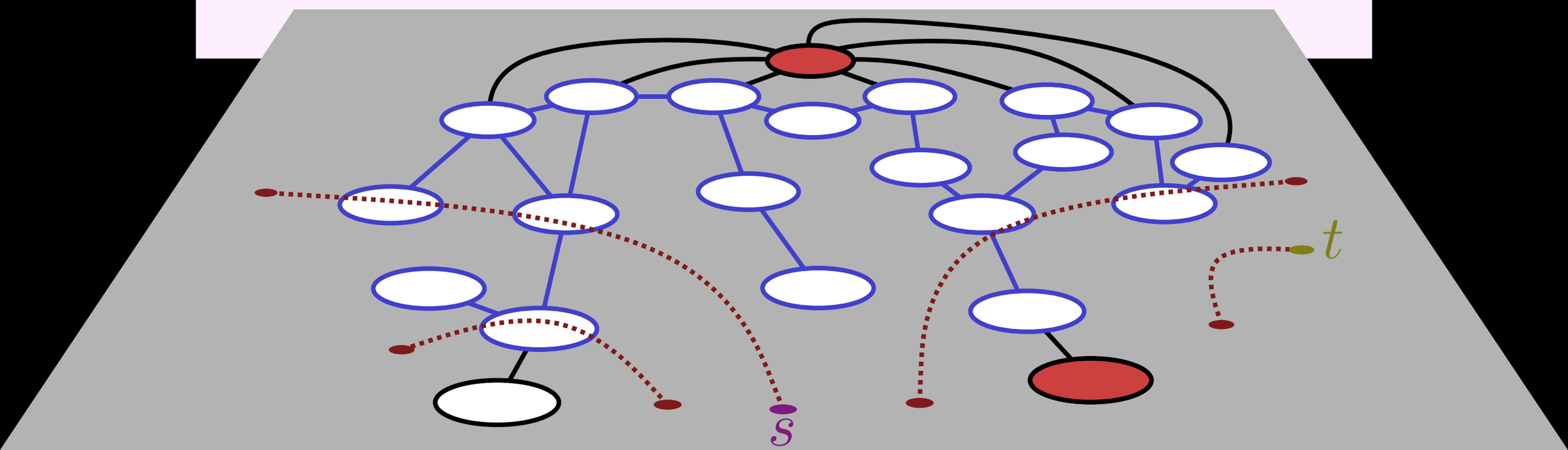


Consider the intersection graph.

Add special *terminal* vertices for pieces of boundary (at most $O(r)$).

We want to *cut* this graph "along the dotted lines".

OBSERVATION: This is the same as cutting between all pairs of terminal vertices that are separated by at least one dotted line.

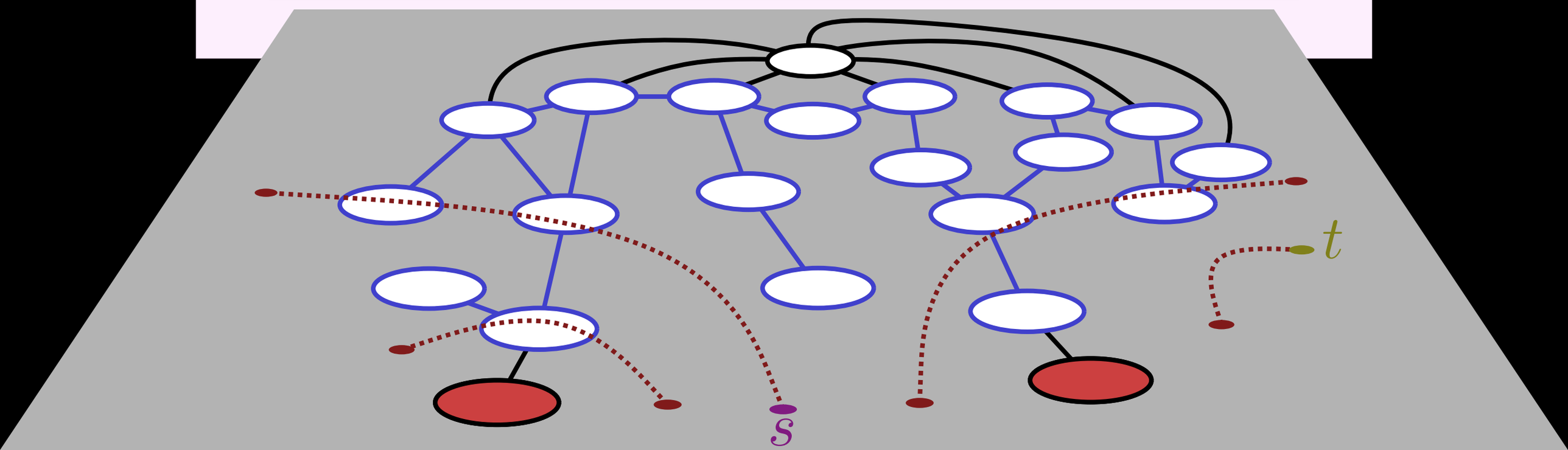


Consider the intersection graph.

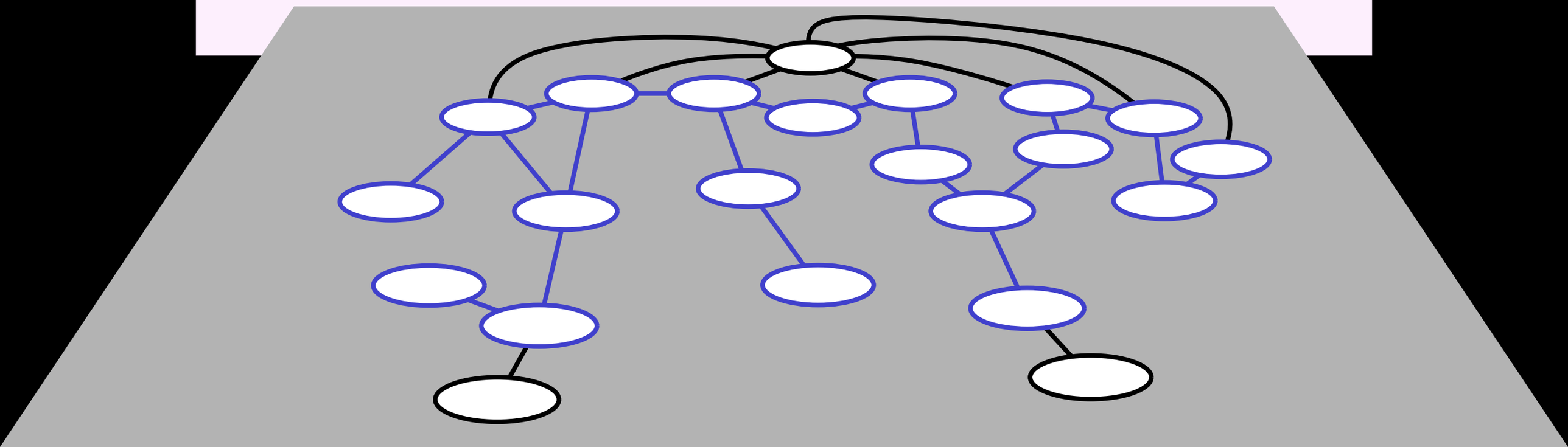
Add special *terminal* vertices for pieces of boundary (at most $O(r)$).

We want to *cut* this graph "along the dotted lines".

OBSERVATION: This is the same as cutting between all pairs of terminal vertices that are separated by at least one dotted line.

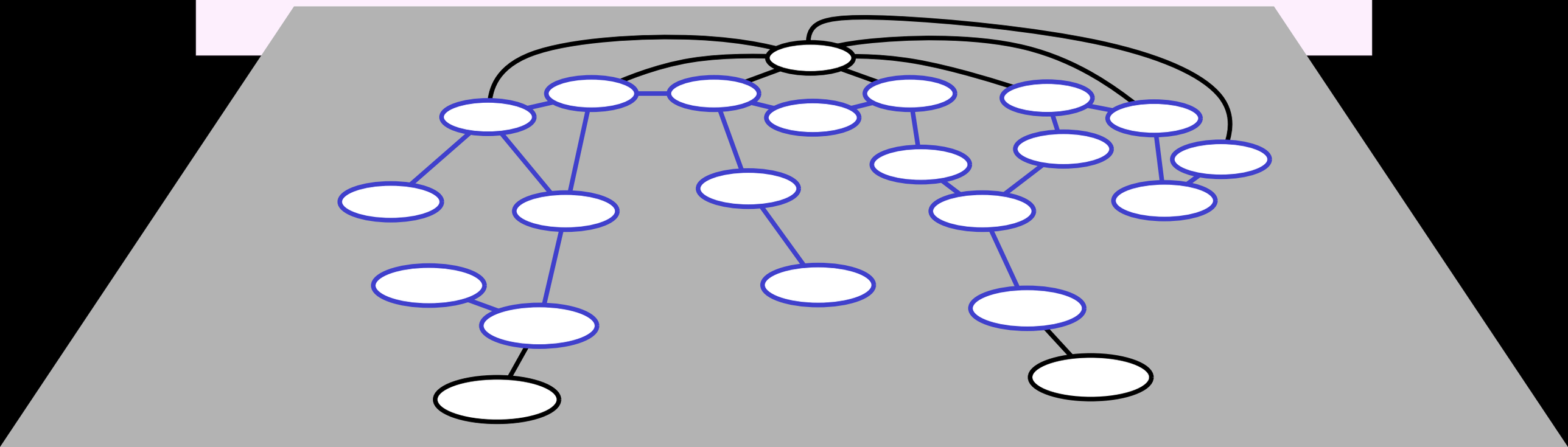


This is known as *vertex multicut*.



This is known as *vertex multicut*.

PROBLEM: Given a graph $G = (V, E)$ and a set of *forbidden pairs* of vertices L, compute the smallest subset $D \subset V$ such that all pairs in L are disconnected in $G \setminus D$.

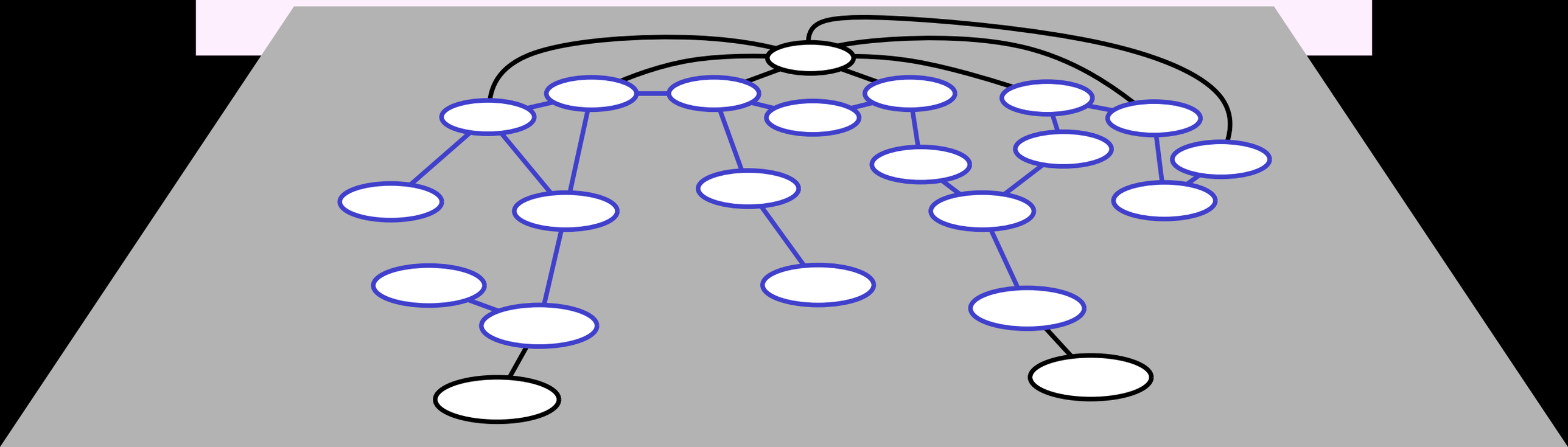


This is known as *vertex multicut*.

PROBLEM: Given a graph $G = (V, E)$ and a set of *forbidden pairs* of vertices L, compute the smallest subset $D \subset V$ such that all pairs in L are disconnected in $G \setminus D$.

An optimal vertex multicut can be computed in $O(2^{f(|D|,|L|)}n^3)$ time.

(Xiao, 2010)

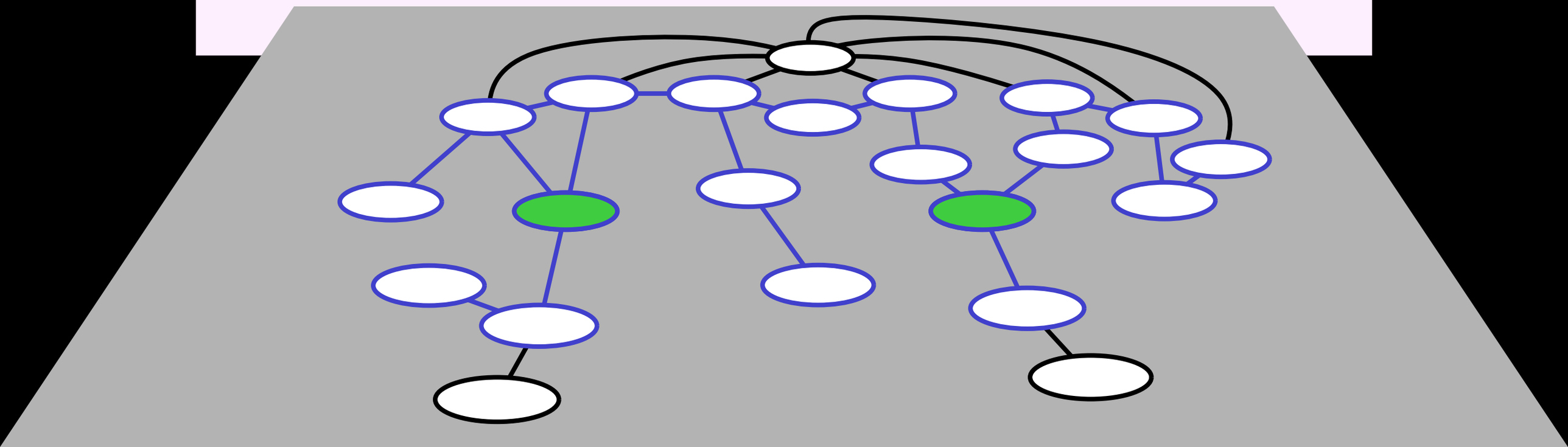


This is known as *vertex multicut*.

PROBLEM: Given a graph $G = (V, E)$ and a set of *forbidden pairs* of vertices L, compute the smallest subset $D \subset V$ such that all pairs in L are disconnected in $G \setminus D$.

An optimal vertex multicut can be computed in $O(2^{f(|D|,|L|)}n^3)$ time.

(Xiao, 2010)

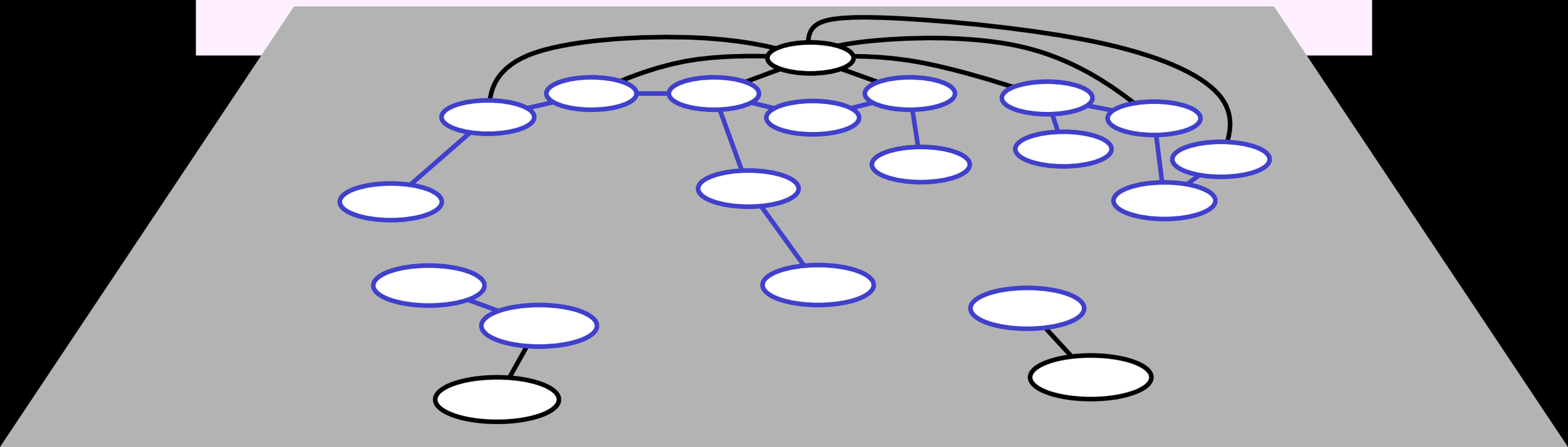


This is known as *vertex multicut*.

PROBLEM: Given a graph $G = (V, E)$ and a set of *forbidden pairs* of vertices L, compute the smallest subset $D \subset V$ such that all pairs in L are disconnected in $G \setminus D$.

An optimal vertex multicut can be computed in $O(2^{f(|D|,|L|)}n^3)$ time.

(Xiao, 2010)

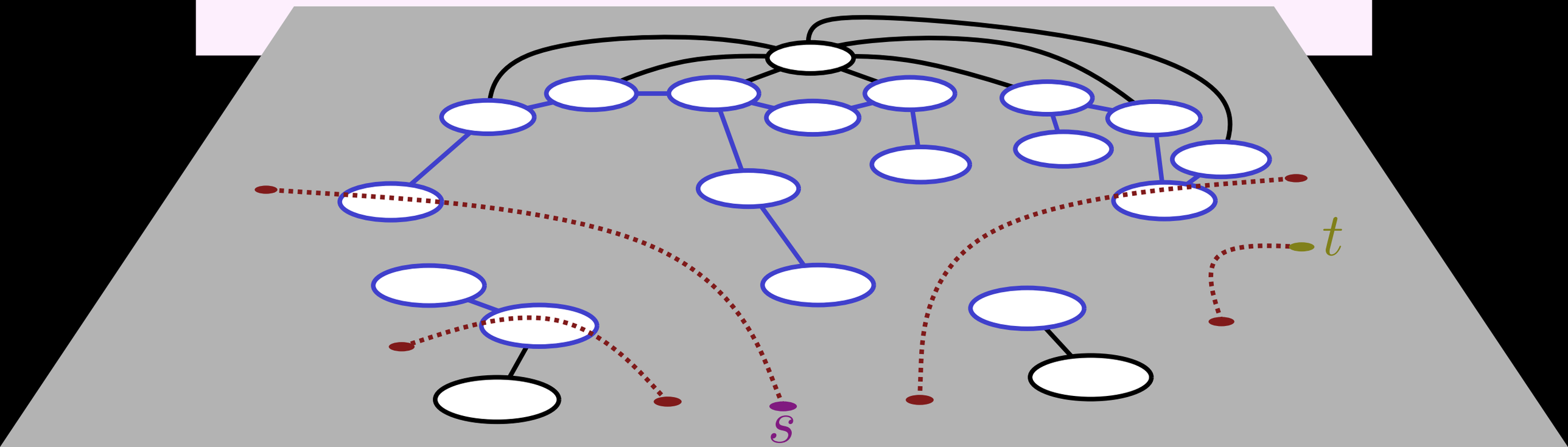


This is known as *vertex multicut*.

PROBLEM: Given a graph $G = (V, E)$ and a set of *forbidden pairs* of vertices L, compute the smallest subset $D \subset V$ such that all pairs in L are disconnected in $G \setminus D$.

An optimal vertex multicut can be computed in $O(2^{f(|D|,|L|)}n^3)$ time.

(Xiao, 2010)

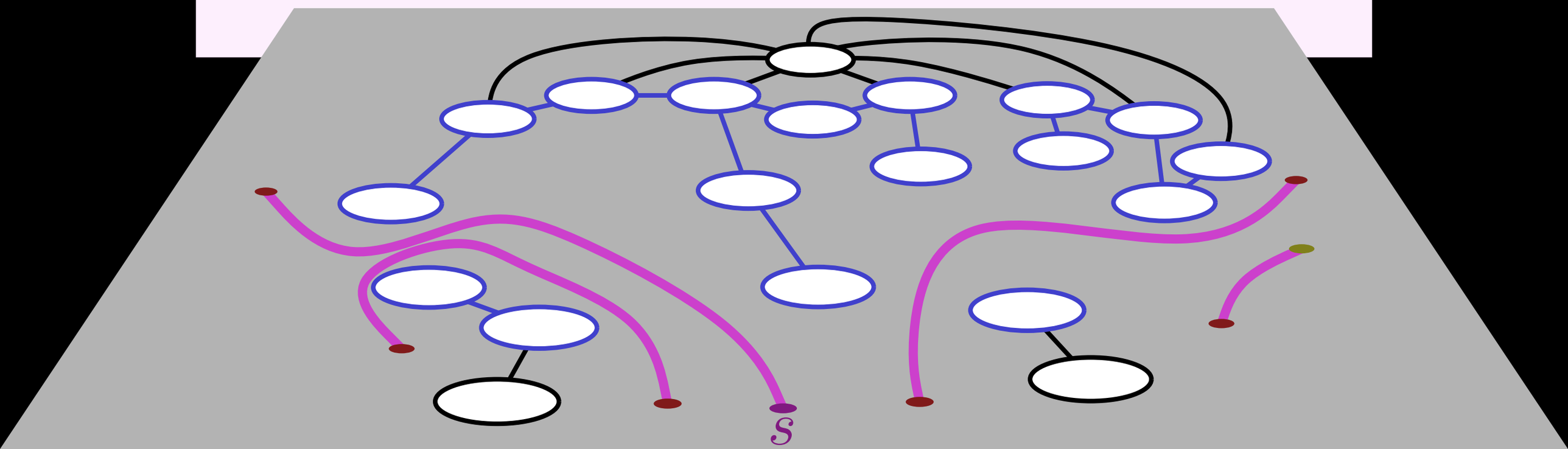


This is known as *vertex multicut*.

PROBLEM: Given a graph $G = (V, E)$ and a set of *forbidden pairs* of vertices L, compute the smallest subset $D \subset V$ such that all pairs in L are disconnected in $G \setminus D$.

An optimal vertex multicut can be computed in $O(2^{f(|D|,|L|)}n^3)$ time.

(Xiao, 2010)

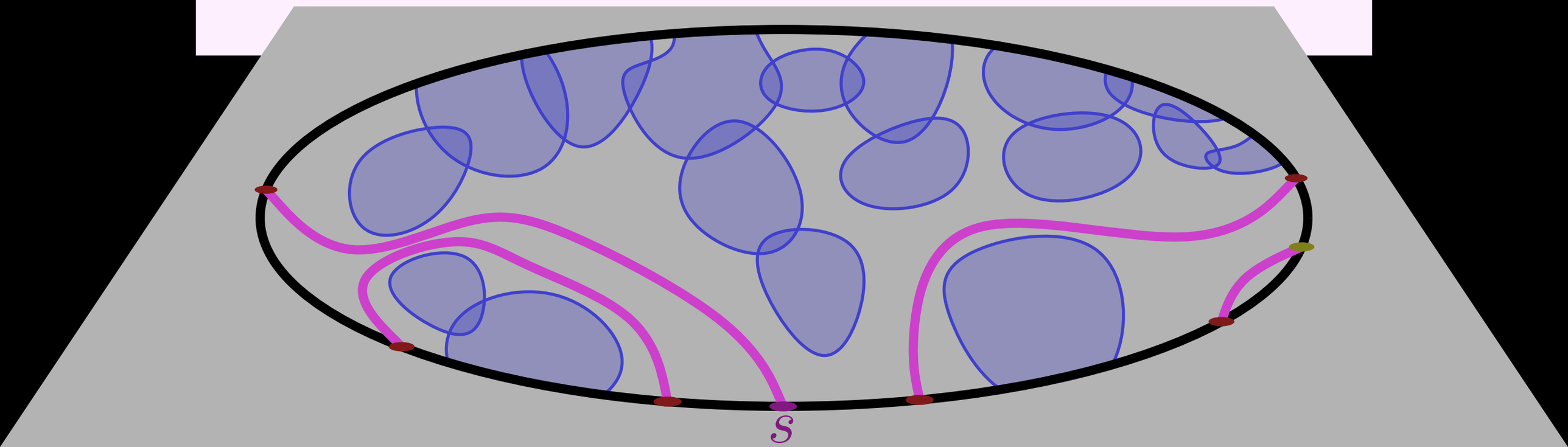


This is known as *vertex multicut*.

PROBLEM: Given a graph $G = (V, E)$ and a set of *forbidden pairs* of vertices L, compute the smallest subset $D \subset V$ such that all pairs in L are disconnected in $G \setminus D$.

An optimal vertex multicut can be computed in $O(2^{f(|D|,|L|)}n^3)$ time.

(Xiao, 2010)

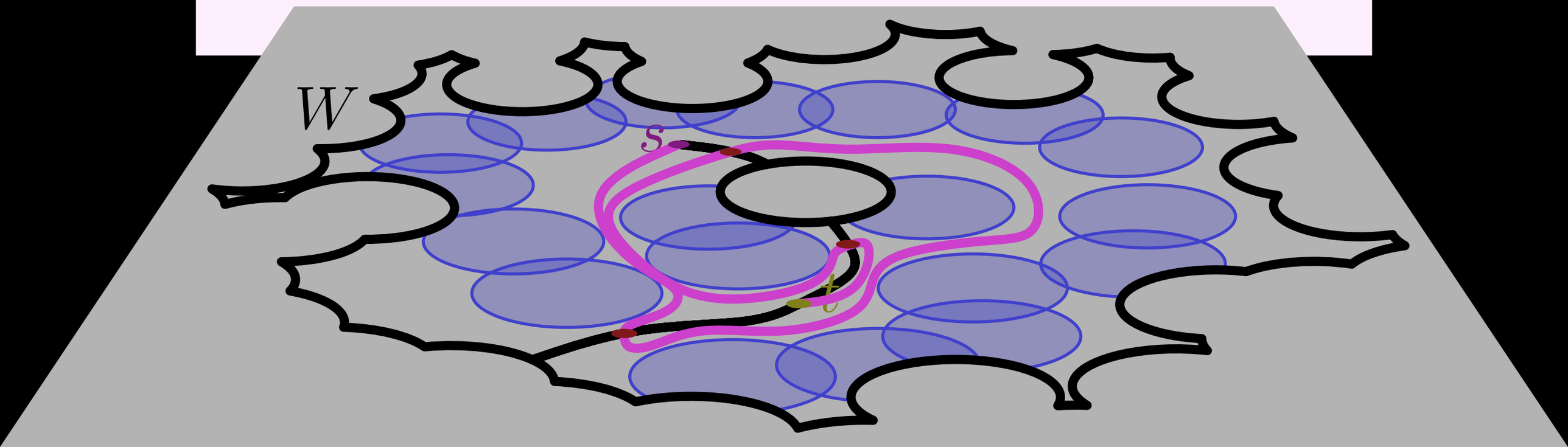


This is known as *vertex multicut*.

PROBLEM: Given a graph $G = (V, E)$ and a set of *forbidden pairs* of vertices L, compute the smallest subset $D \subset V$ such that all pairs in L are disconnected in $G \setminus D$.

An optimal vertex multicut can be computed in $O(2^{f(|D|,|L|)}n^3)$ time.

(Xiao, 2010)

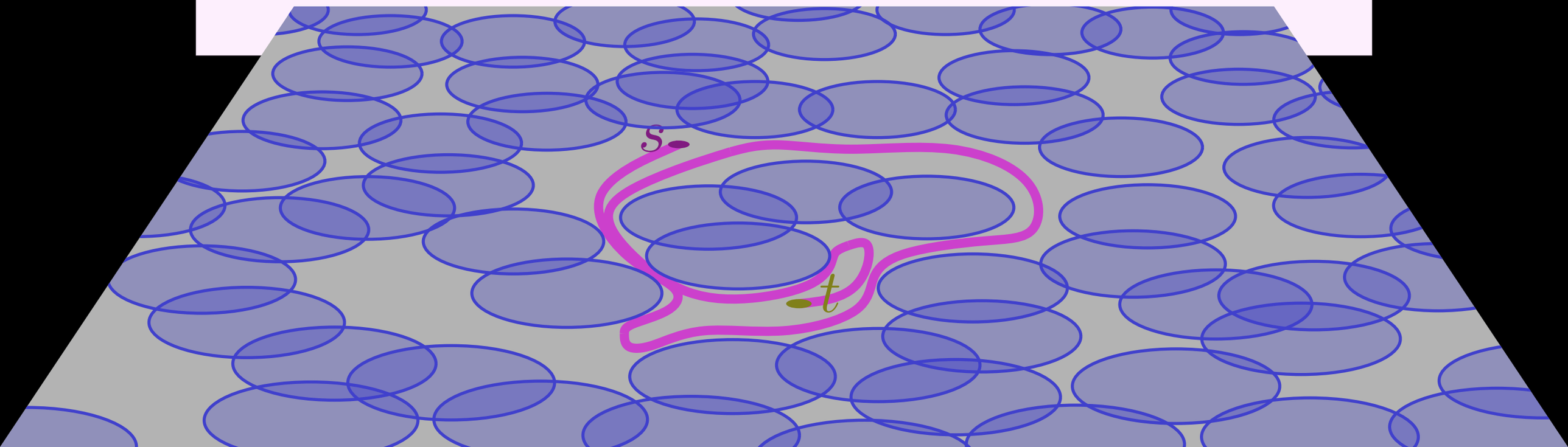


This is known as *vertex multicut*.

PROBLEM: Given a graph $G = (V, E)$ and a set of *forbidden pairs* of vertices L, compute the smallest subset $D \subset V$ such that all pairs in L are disconnected in $G \setminus D$.

An optimal vertex multicut can be computed in $O(2^{f(|D|,|L|)}n^3)$ time.

(Xiao, 2010)

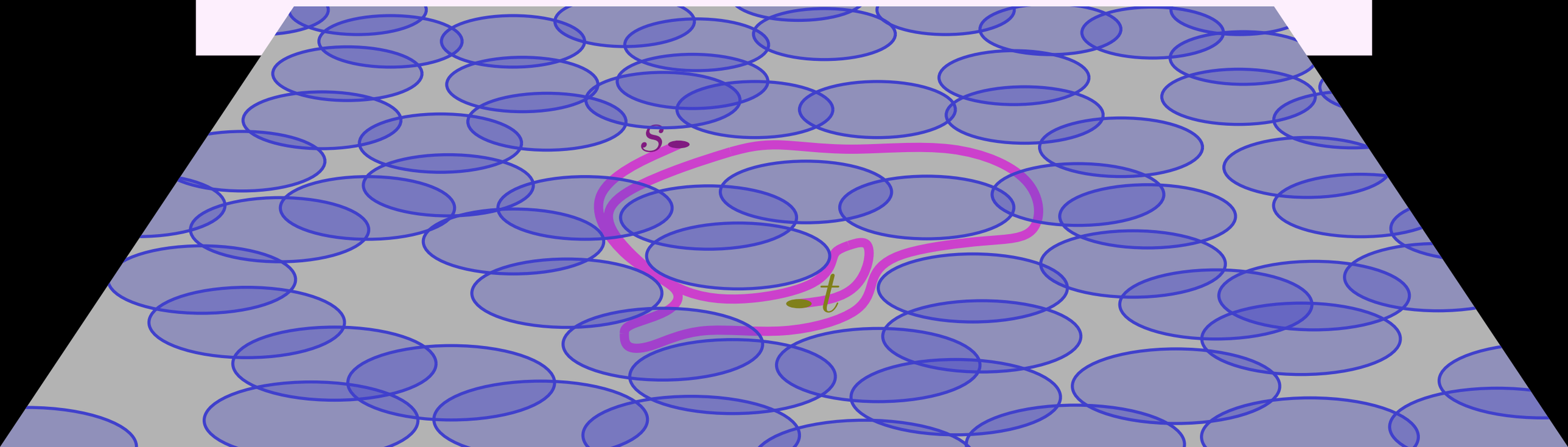


This is known as *vertex multicut*.

PROBLEM: Given a graph $G = (V, E)$ and a set of *forbidden pairs* of vertices L, compute the smallest subset $D \subset V$ such that all pairs in L are disconnected in $G \setminus D$.

An optimal vertex multicut can be computed in $O(2^{f(|D|,|L|)}n^3)$ time.

(Xiao, 2010)

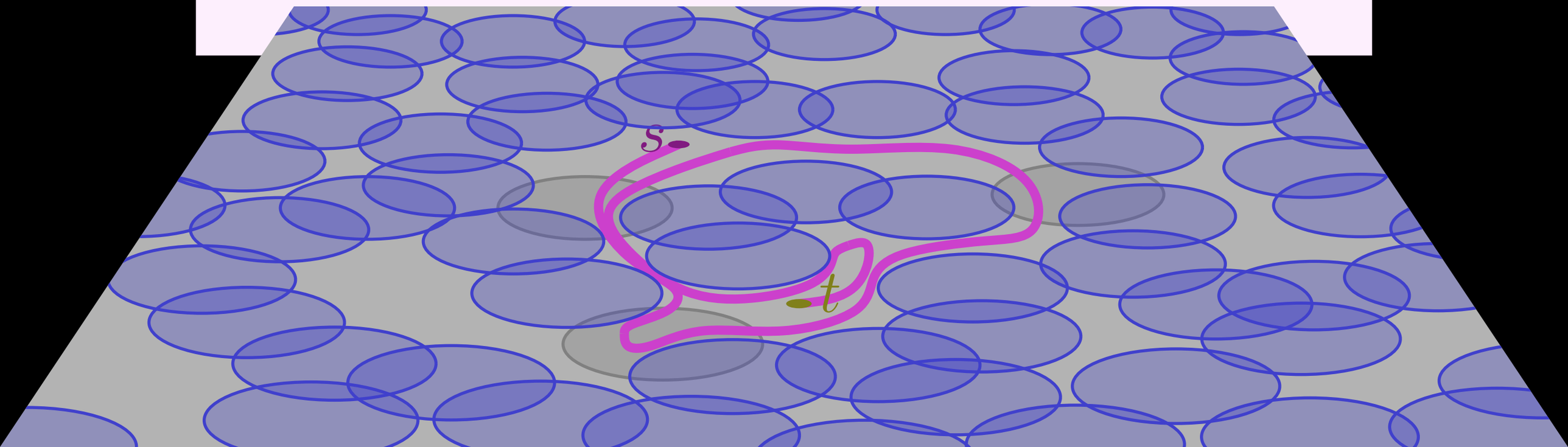


This is known as *vertex multicut*.

PROBLEM: Given a graph $G = (V, E)$ and a set of *forbidden pairs* of vertices L, compute the smallest subset $D \subset V$ such that all pairs in L are disconnected in $G \setminus D$.

An optimal vertex multicut can be computed in $O(2^{f(|D|,|L|)}n^3)$ time.

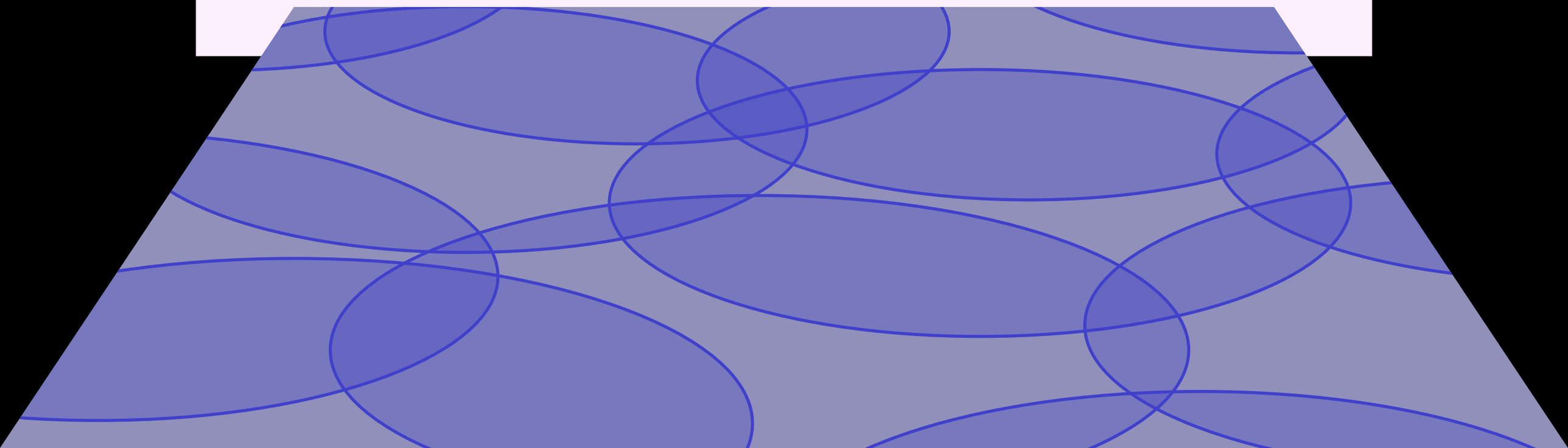
(Xiao, 2010)



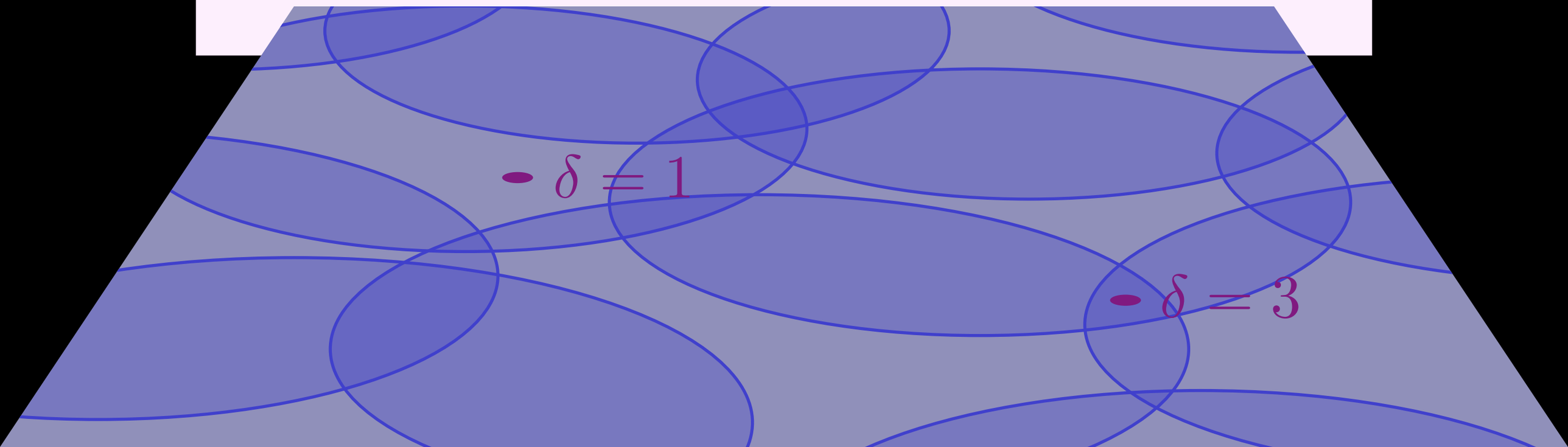
ε-APPROXIMATION ALGORITHM

The *ply* δ of a point is the total number of regions that contain it.

The *ply* δ of a point is the total number of regions that contain it.

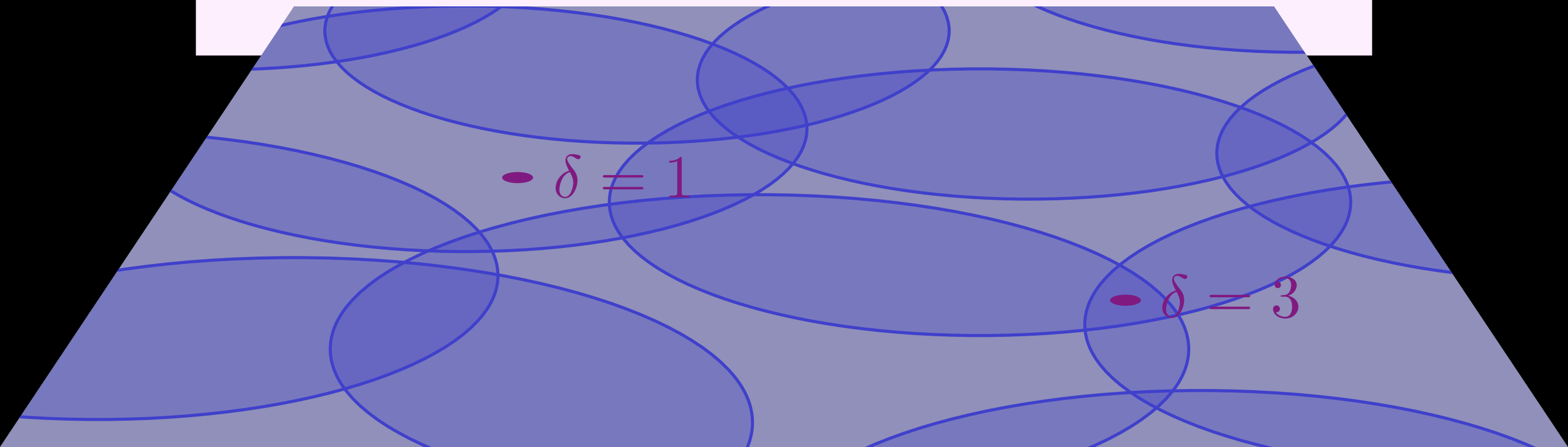


The *ply* δ of a point is the total number of regions that contain it.



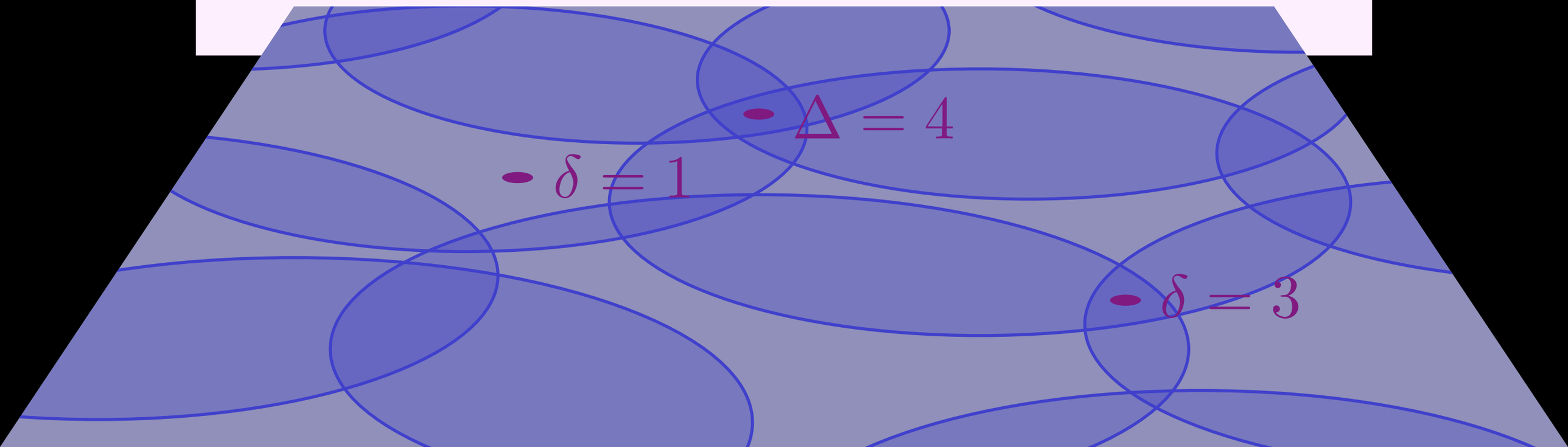
The *ply* δ of a point is the total number of regions that contain it.

The ply Δ of $\mathcal{R}$ is the maximum ply over all points in the plane.



The *ply* δ of a point is the total number of regions that contain it.

The ply Δ of $\mathcal{R}$ is the maximum ply over all points in the plane.



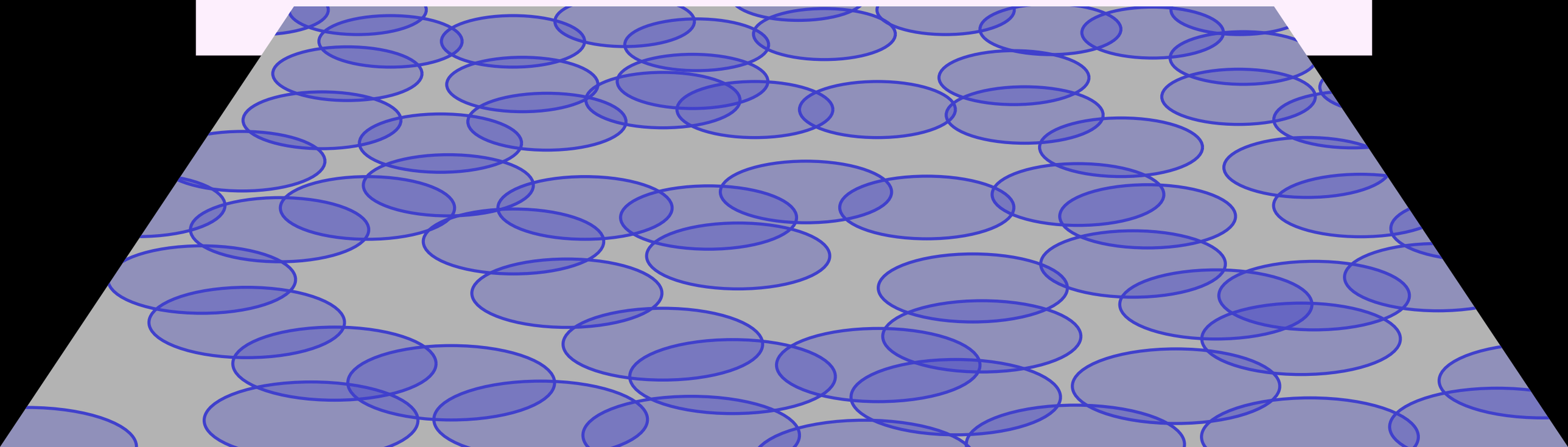
Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

Compute all pairs of resilience at most k exactly, using the FPT algorithm.

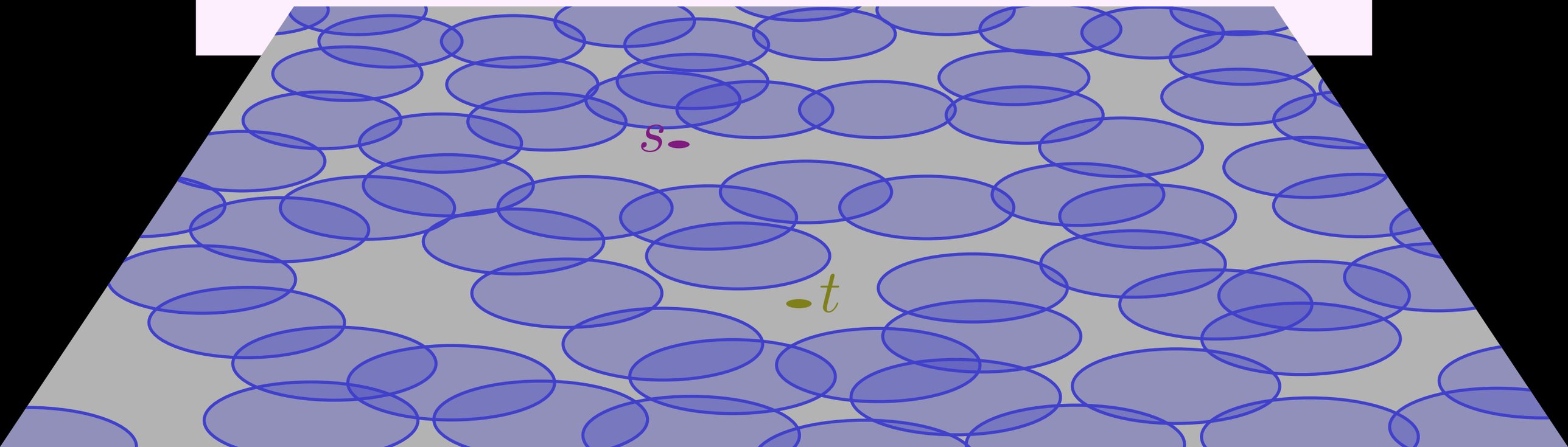
Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

Compute all pairs of resilience at most k exactly, using the FPT algorithm.



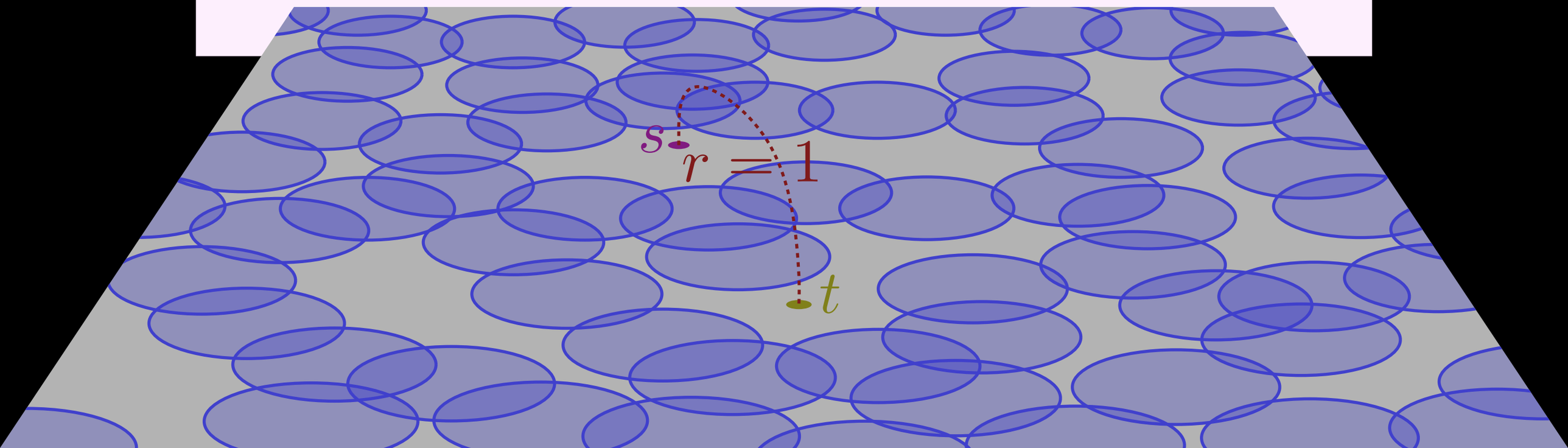
Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

Compute all pairs of resilience at most k exactly, using the FPT algorithm.



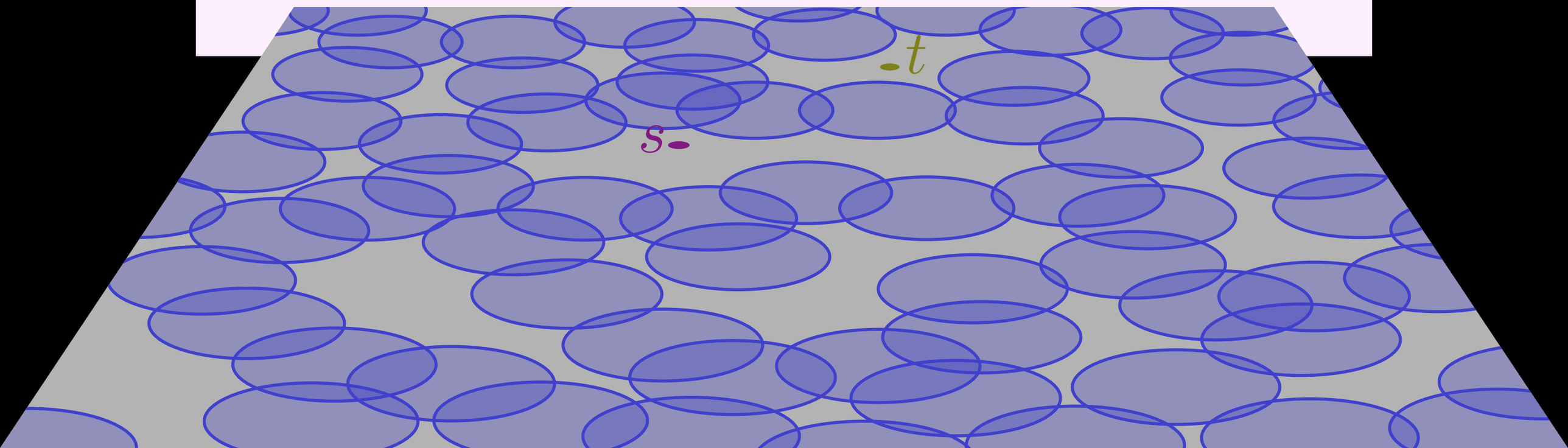
Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

Compute all pairs of resilience at most k exactly, using the FPT algorithm.



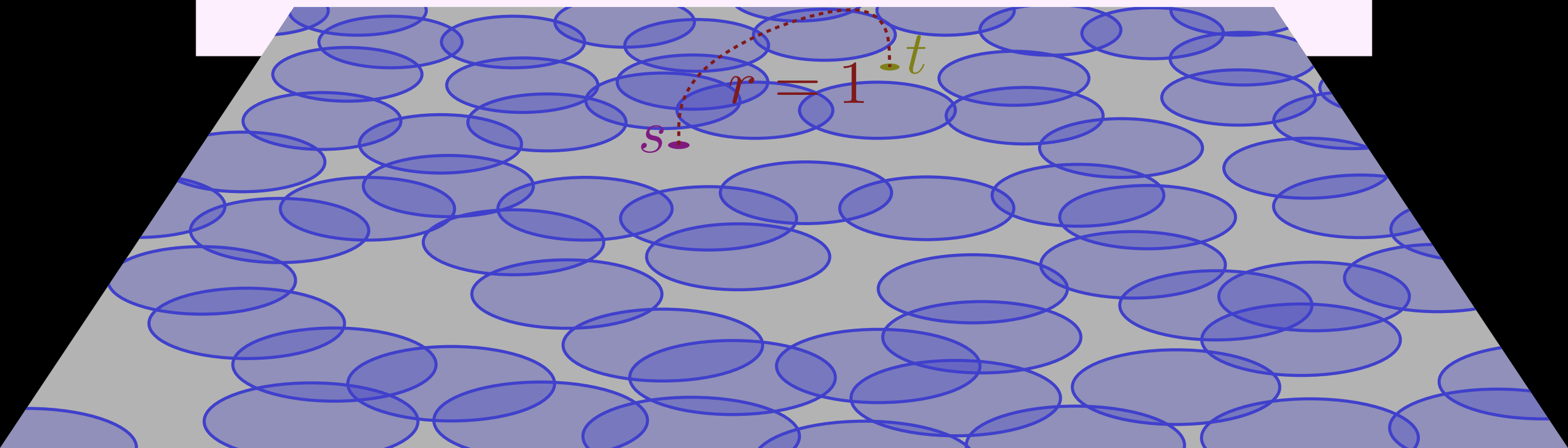
Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

Compute all pairs of resilience at most k exactly, using the FPT algorithm.



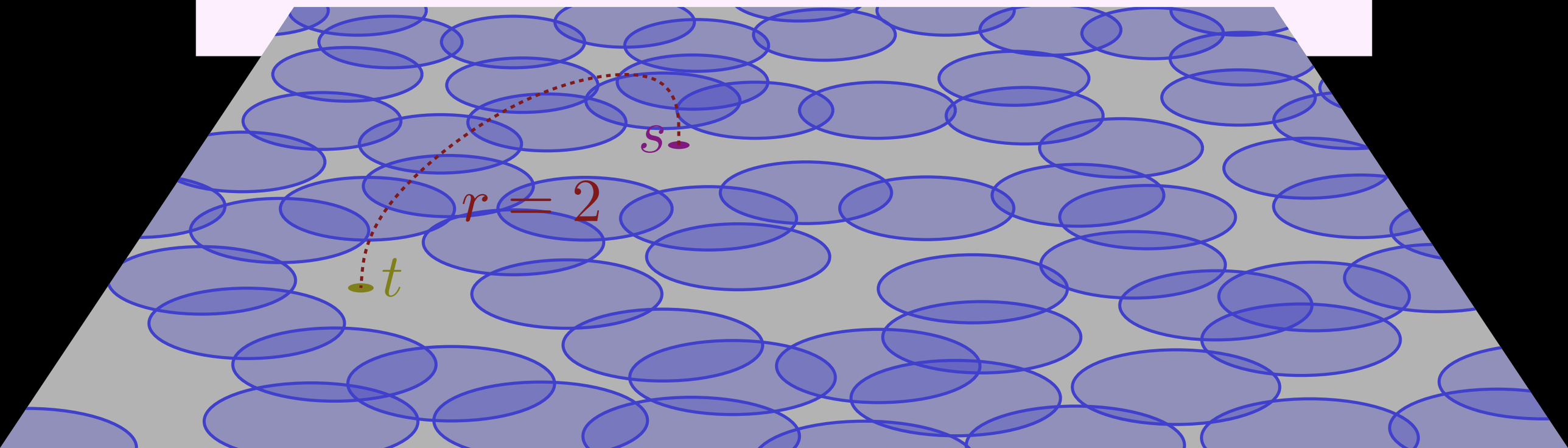
Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

Compute all pairs of resilience at most k exactly, using the FPT algorithm.



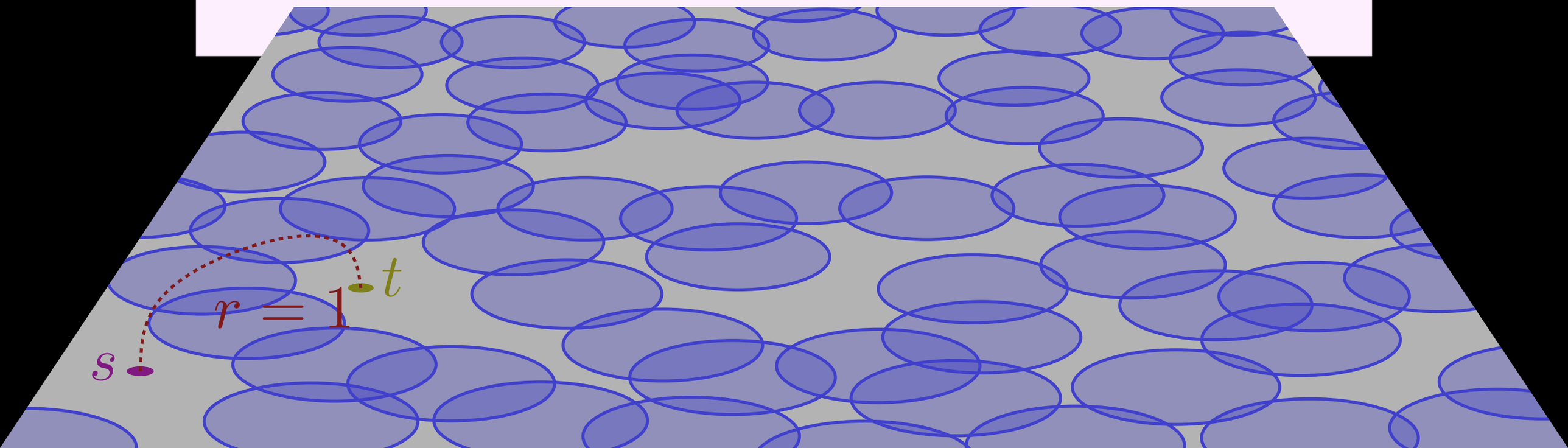
Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

Compute all pairs of resilience at most k exactly, using the FPT algorithm.



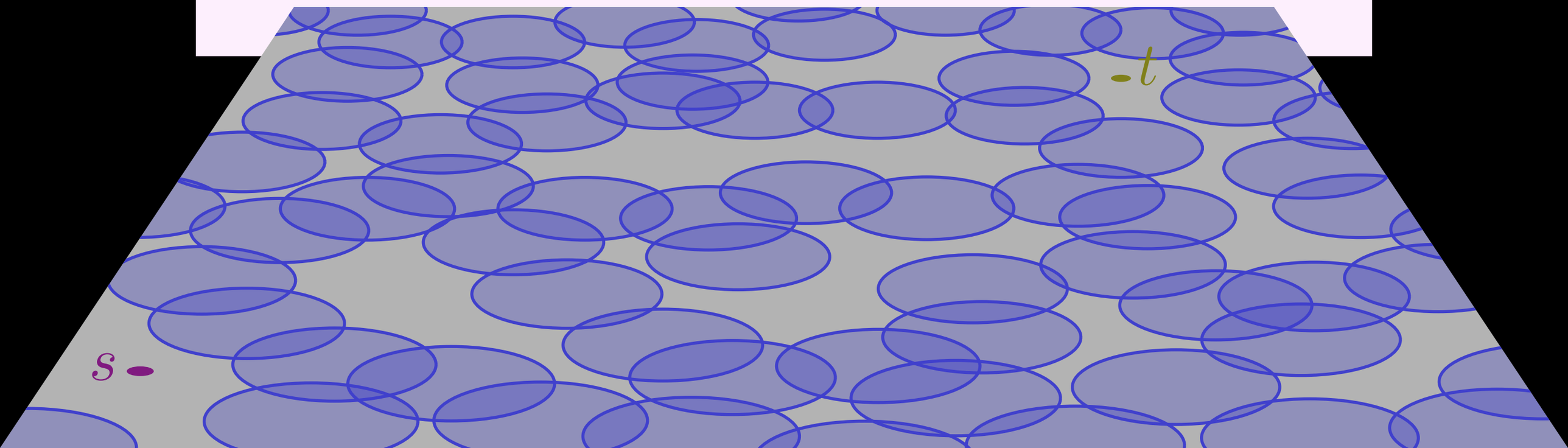
Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

Compute all pairs of resilience at most k exactly, using the FPT algorithm.



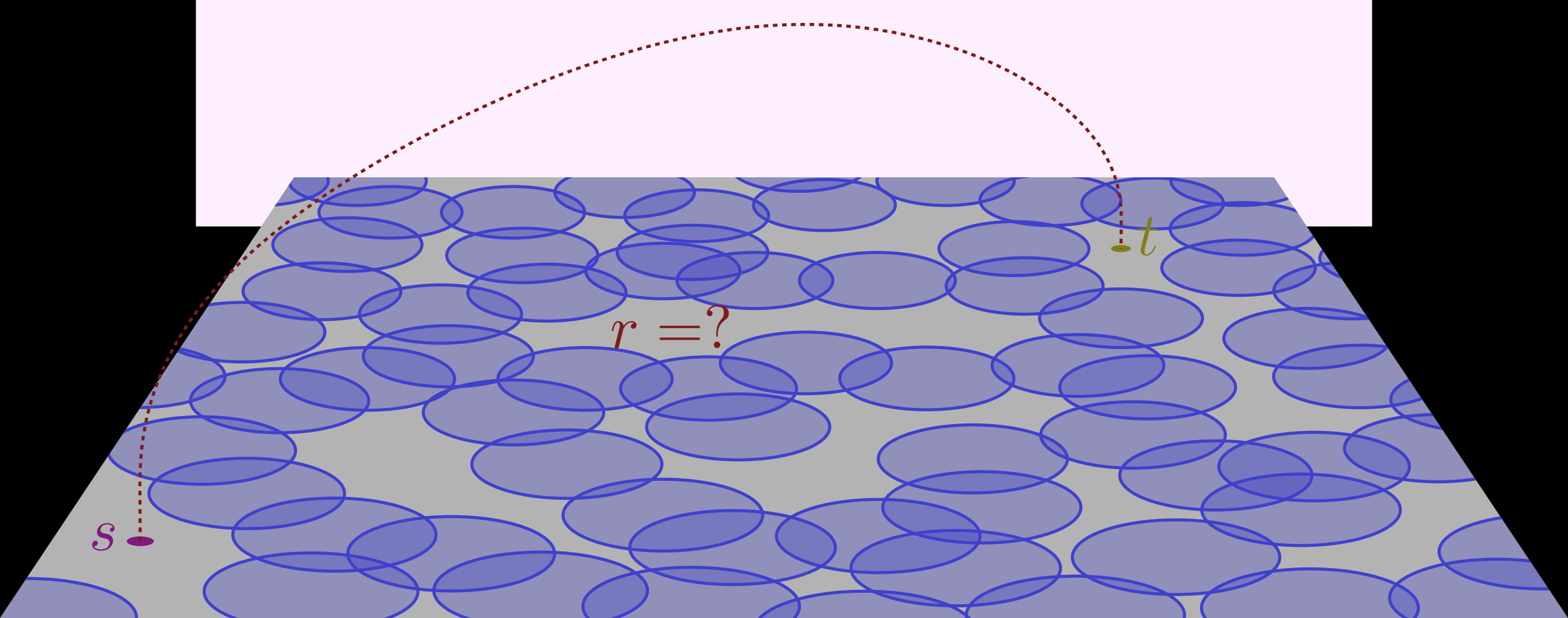
Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

Compute all pairs of resilience at most k exactly, using the FPT algorithm.



Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

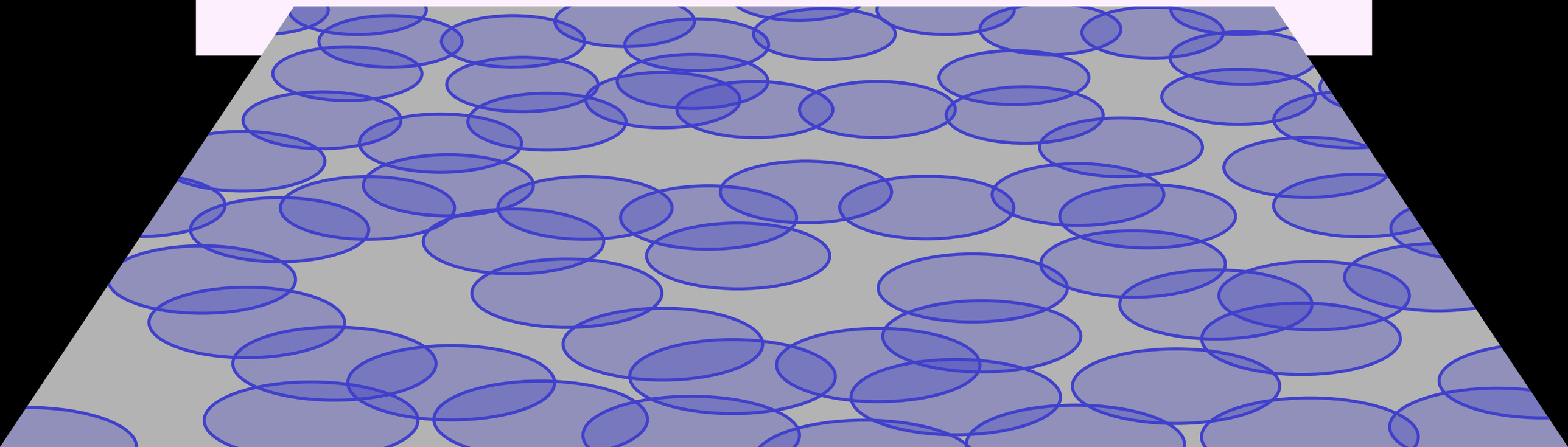
Compute all pairs of resilience at most k exactly, using the FPT algorithm.



Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

Compute all pairs of resilience at most k exactly, using the FPT algorithm.

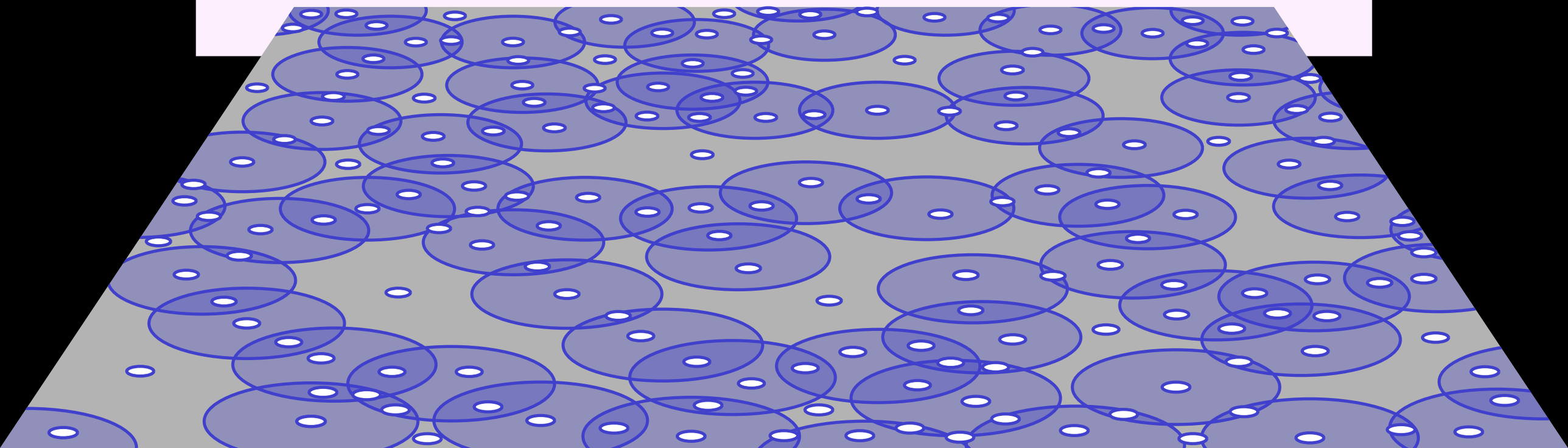
Augment the dual graph of $\mathcal{R}$ with extra edges.



Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

Compute all pairs of resilience at most k exactly, using the FPT algorithm.

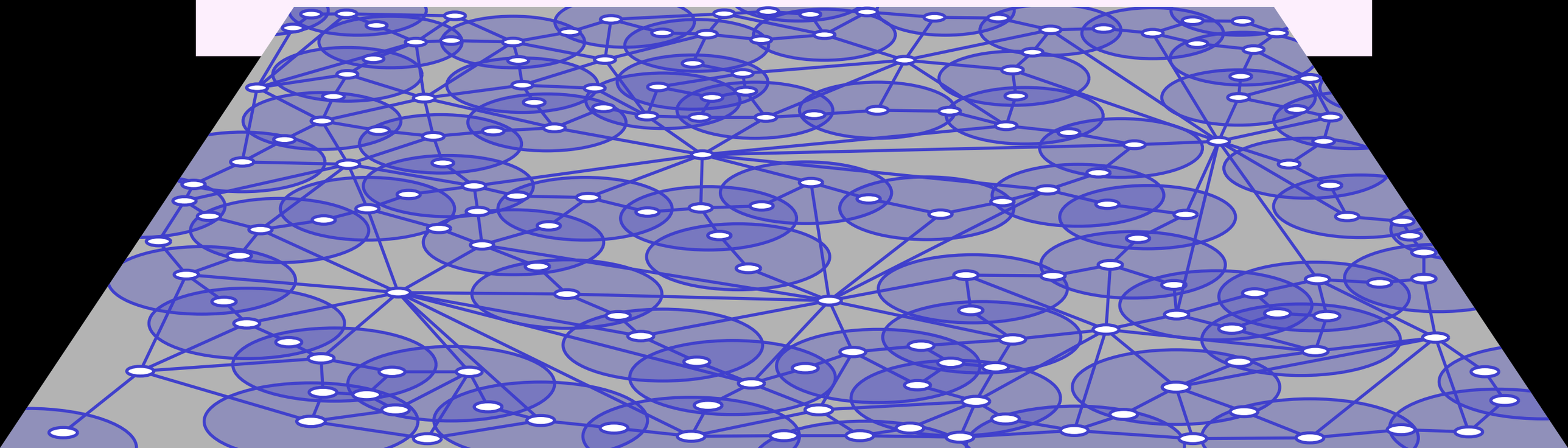
Augment the dual graph of $\mathcal{R}$ with extra edges.



Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

Compute all pairs of resilience at most k exactly, using the FPT algorithm.

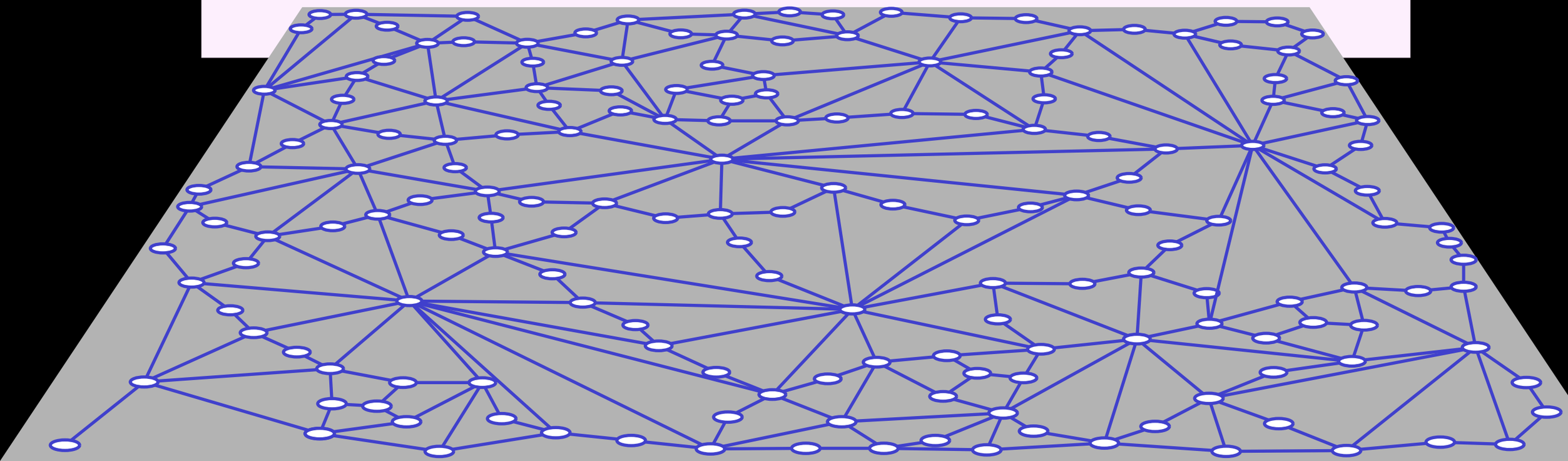
Augment the dual graph of $\mathcal{R}$ with extra edges.



Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

Compute all pairs of resilience at most k exactly, using the FPT algorithm.

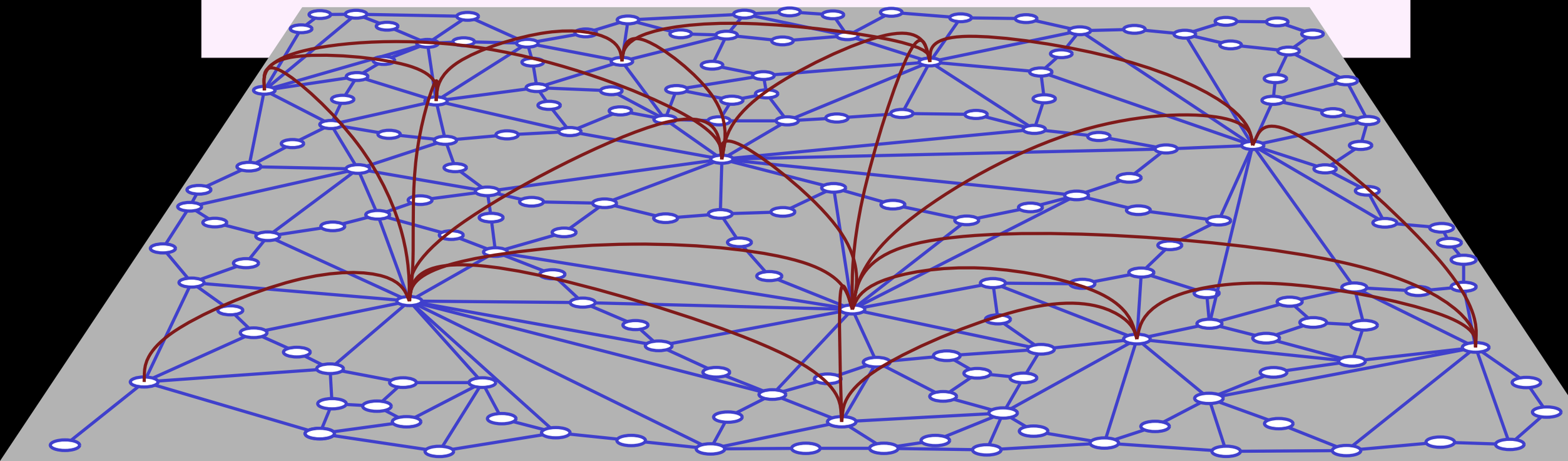
Augment the dual graph of $\mathcal{R}$ with extra edges.



Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

Compute all pairs of resilience at most k exactly, using the FPT algorithm.

Augment the dual graph of $\mathcal{R}$ with extra edges.

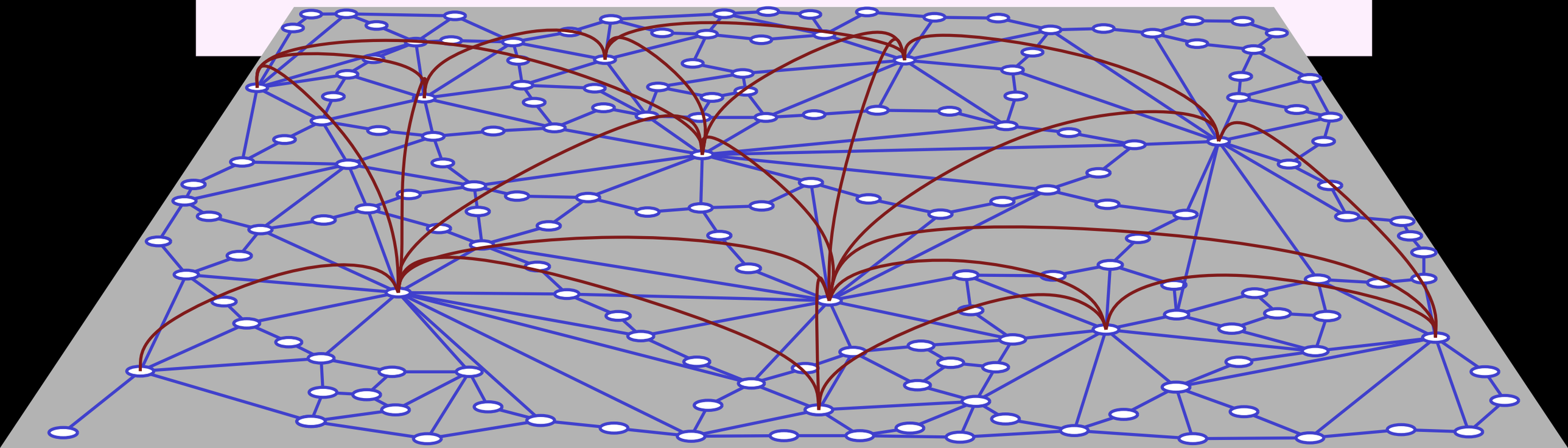


Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

Compute all pairs of resilience at most k exactly, using the FPT algorithm.

Augment the dual graph of $\mathcal{R}$ with extra edges.

Compute the shortest path in the resulting graph.

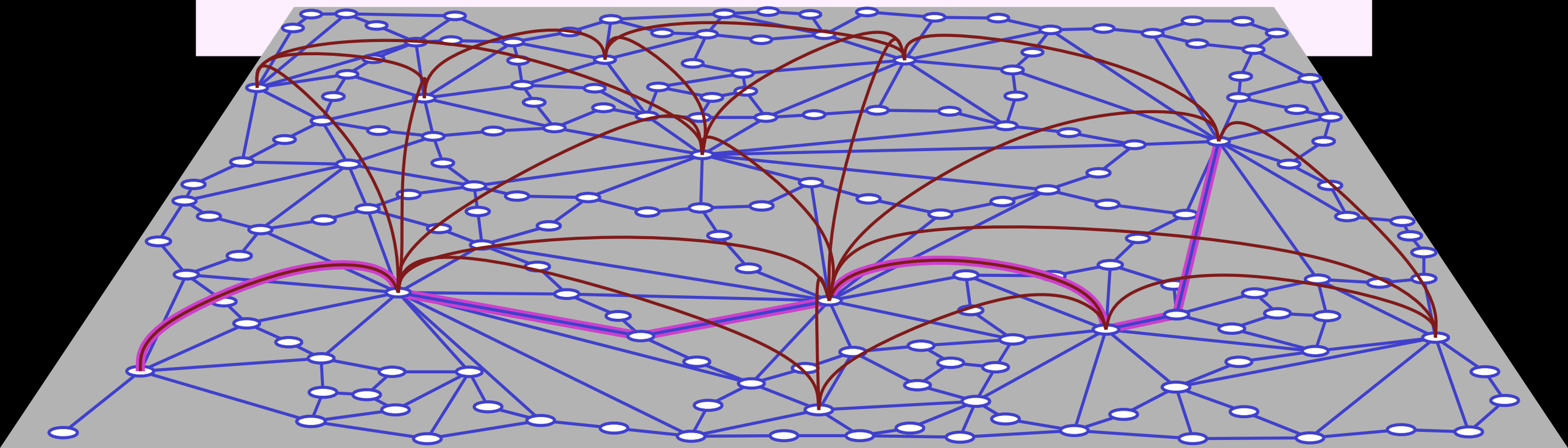


Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

Compute all pairs of resilience at most k exactly, using the FPT algorithm.

Augment the dual graph of $\mathcal{R}$ with extra edges.

Compute the shortest path in the resulting graph.



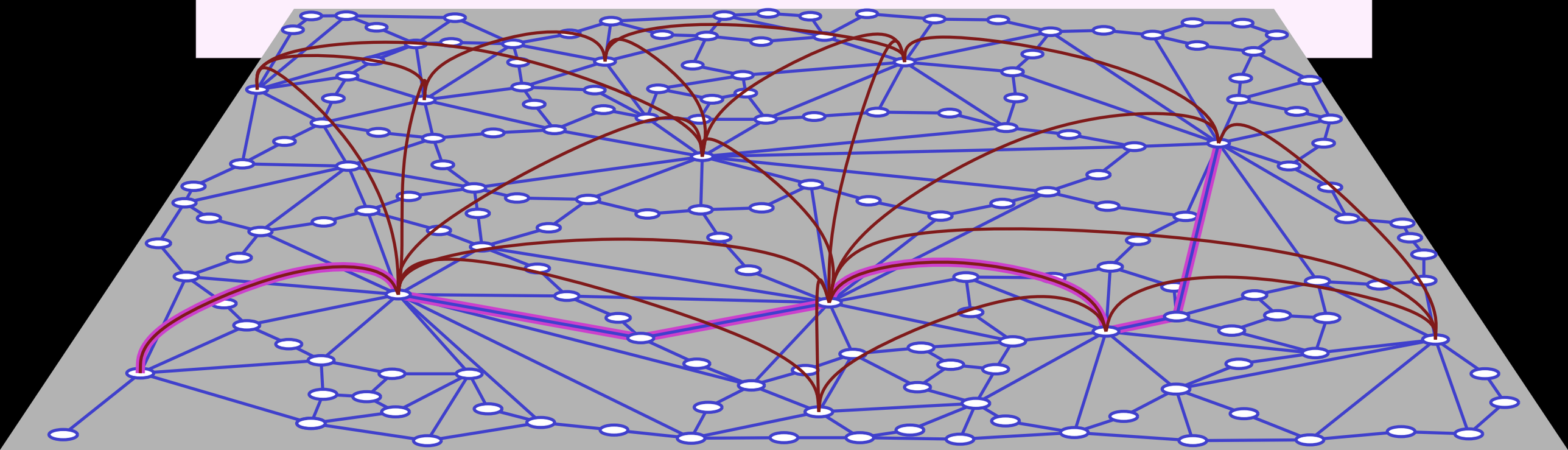
Let $k = \lceil 4\Delta/\varepsilon^2 \rceil$.

Compute all pairs of resilience at most k exactly, using the FPT algorithm.

Augment the dual graph of $\mathcal{R}$ with extra edges.

Compute the shortest path in the resulting graph.

CLAIM: This is a $1 + \varepsilon$ approximation.



CONCLUSIONS

We show that resilience for unit disks is FPT, and leads this to an EPTAS.

We show that resilience for unit disks is FPT, and leads this to an EPTAS.

On the other hand, resilience of many classes of regions is NP-hard.

We show that resilience for unit disks is FPT, and leads this to an EPTAS.

On the other hand, resilience of many classes of regions is NP-hard.

It seems all hardness reductions require regions to cross each other.

We show that resilience for unit disks is FPT, and leads this to an EPTAS.

On the other hand, resilience of many classes of regions is NP-hard.

It seems all hardness reductions require regions to cross each other.

OPEN QUESTION: Can the resilience of unit disks be computed in polynomial time?

THANKS! ;)