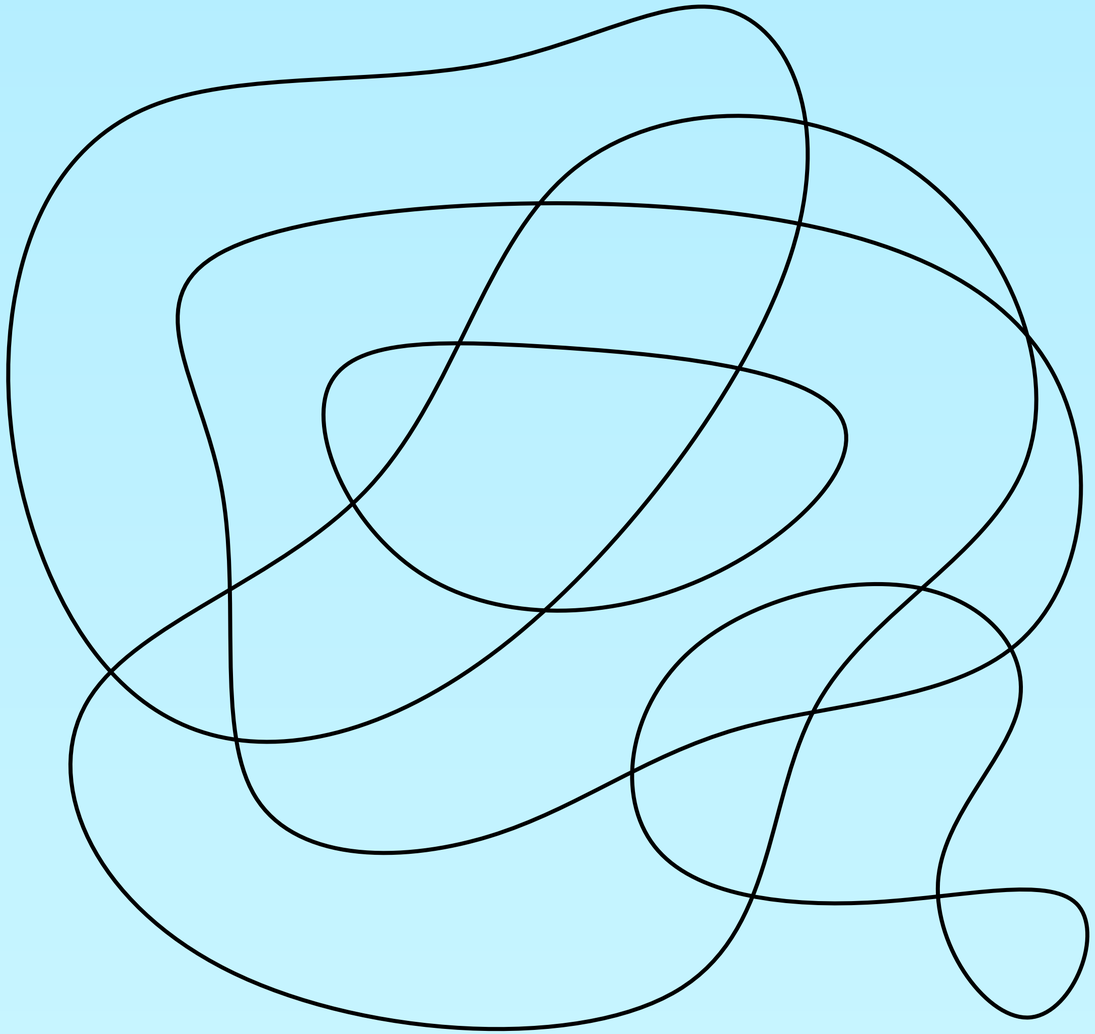
Reconfiguration in Curve Arrangements to Reduce Self-Intersections and Popular Faces

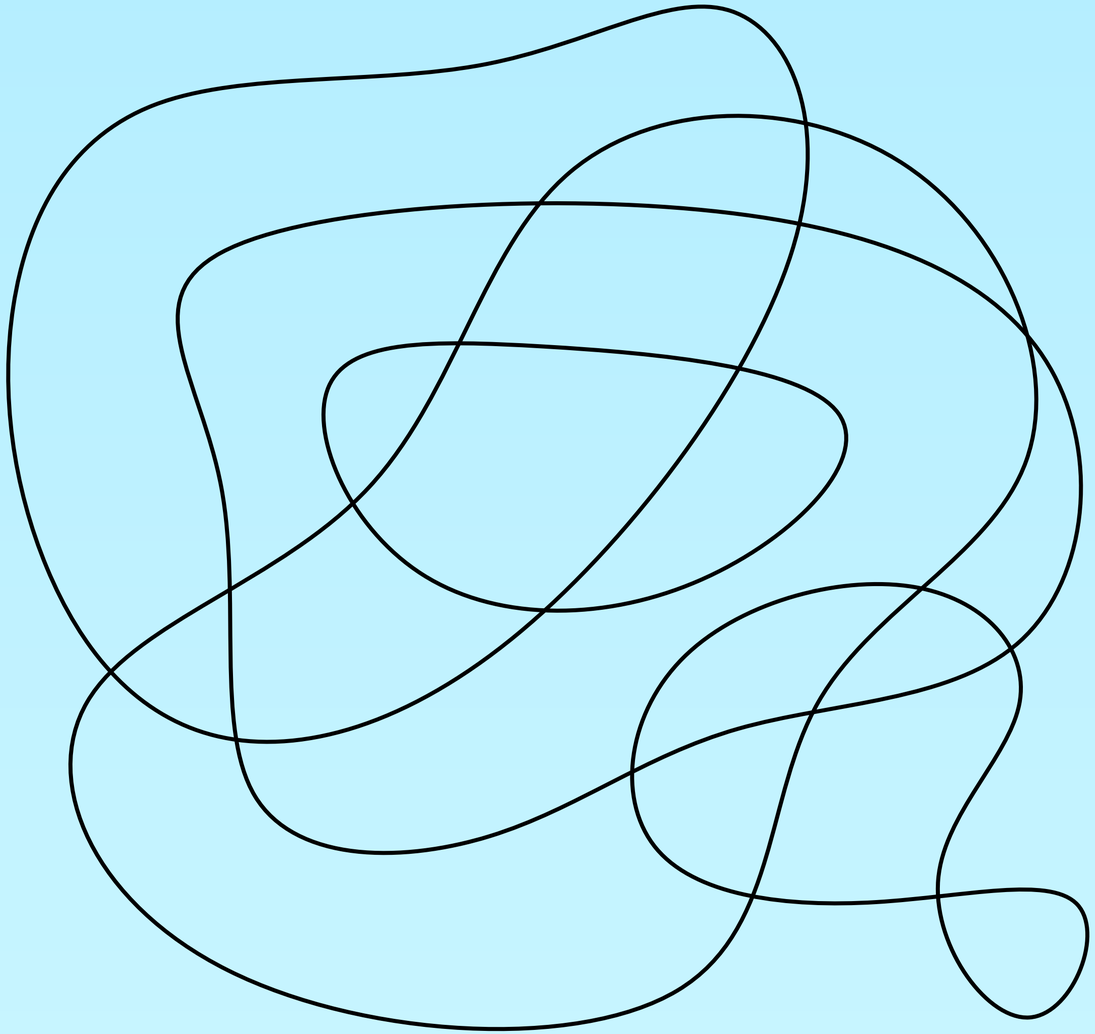
Maarten Löffler
Utrecht University / Tulane University

Joint work with

Florestan Brunck
Hsien-Chih Chang
Tim Ophelders
Lena Schlipf

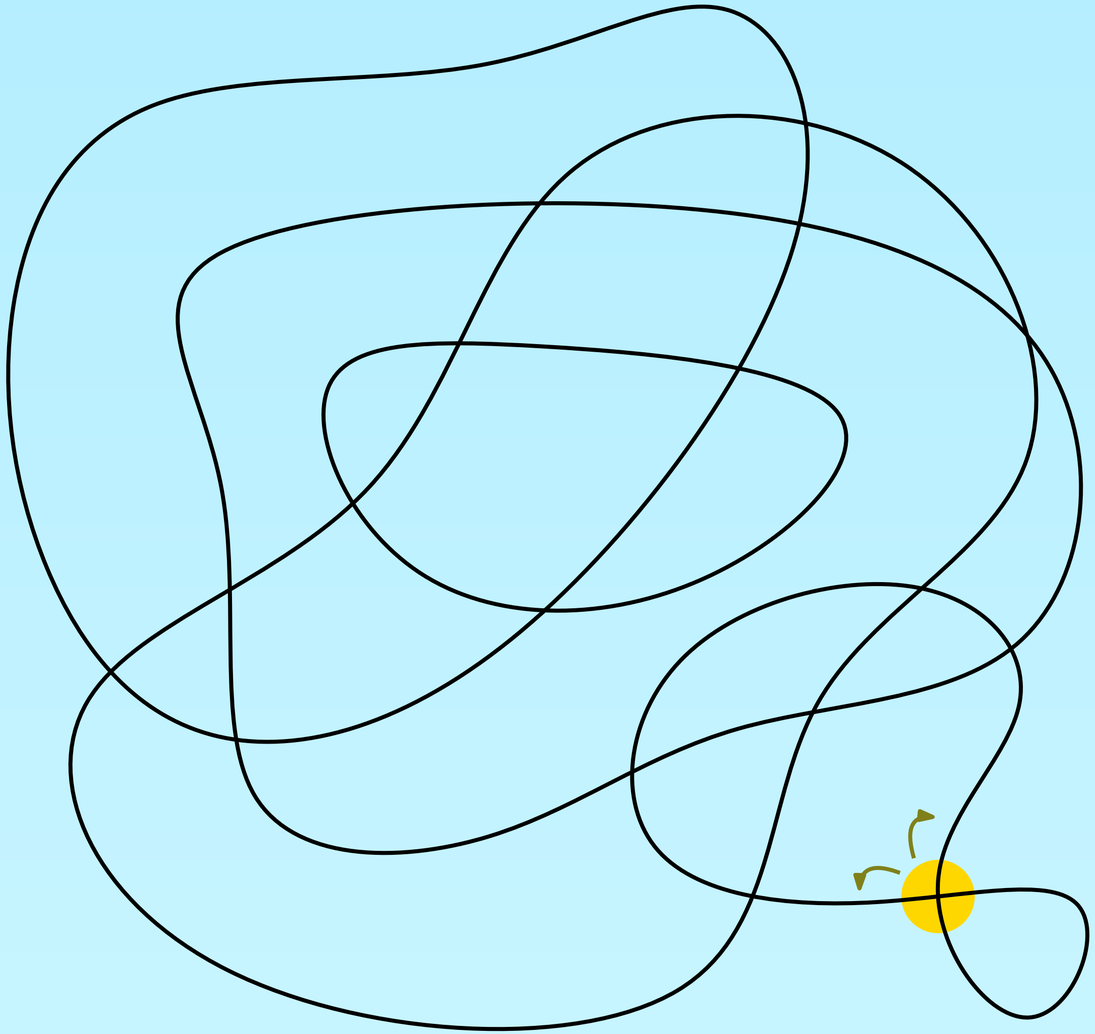


Consider an arrangement of closed curves in the plane.



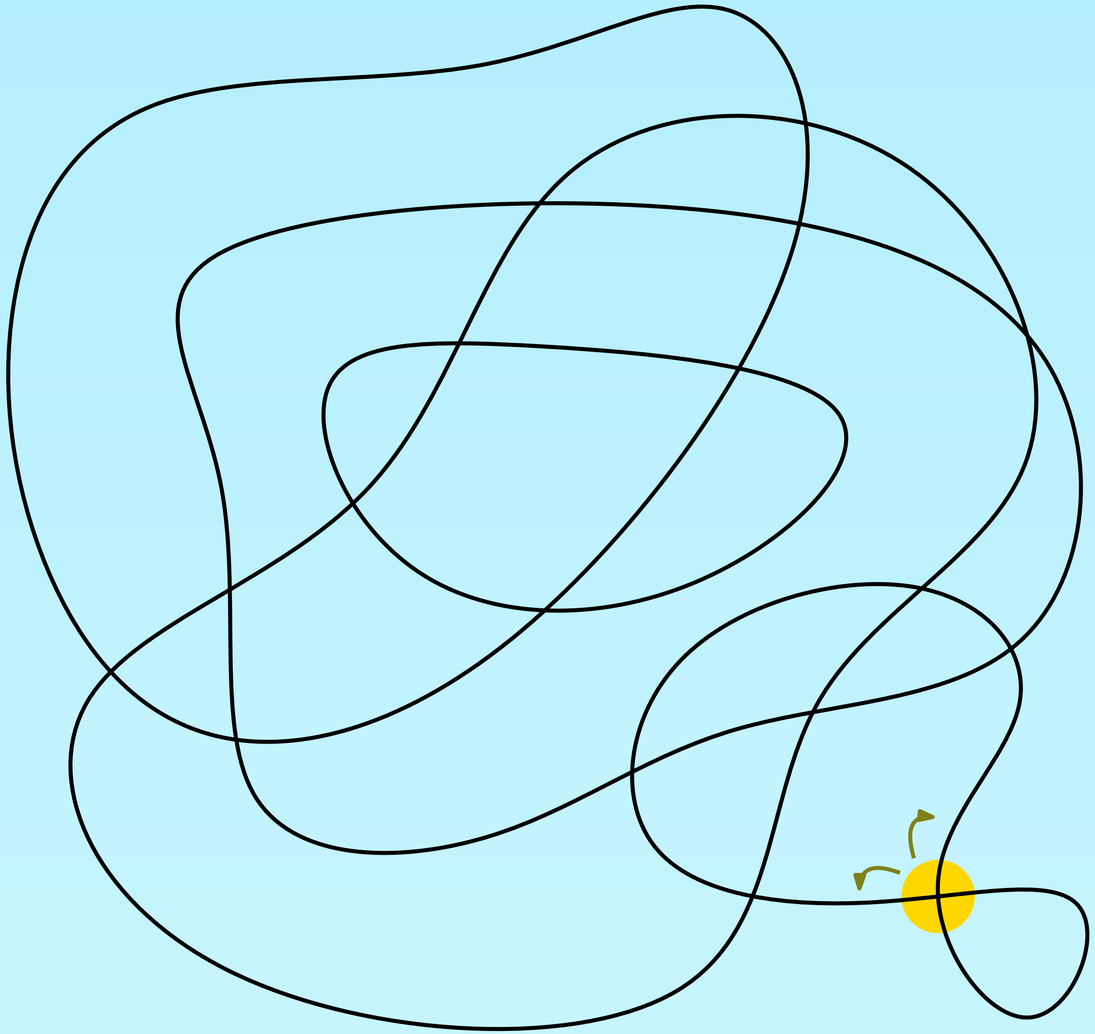
Consider an arrangement of closed curves in the plane.

A vertex of the arrangement that is incident to the same curve twice is a self-intersection.



Consider an arrangement of closed curves in the plane.

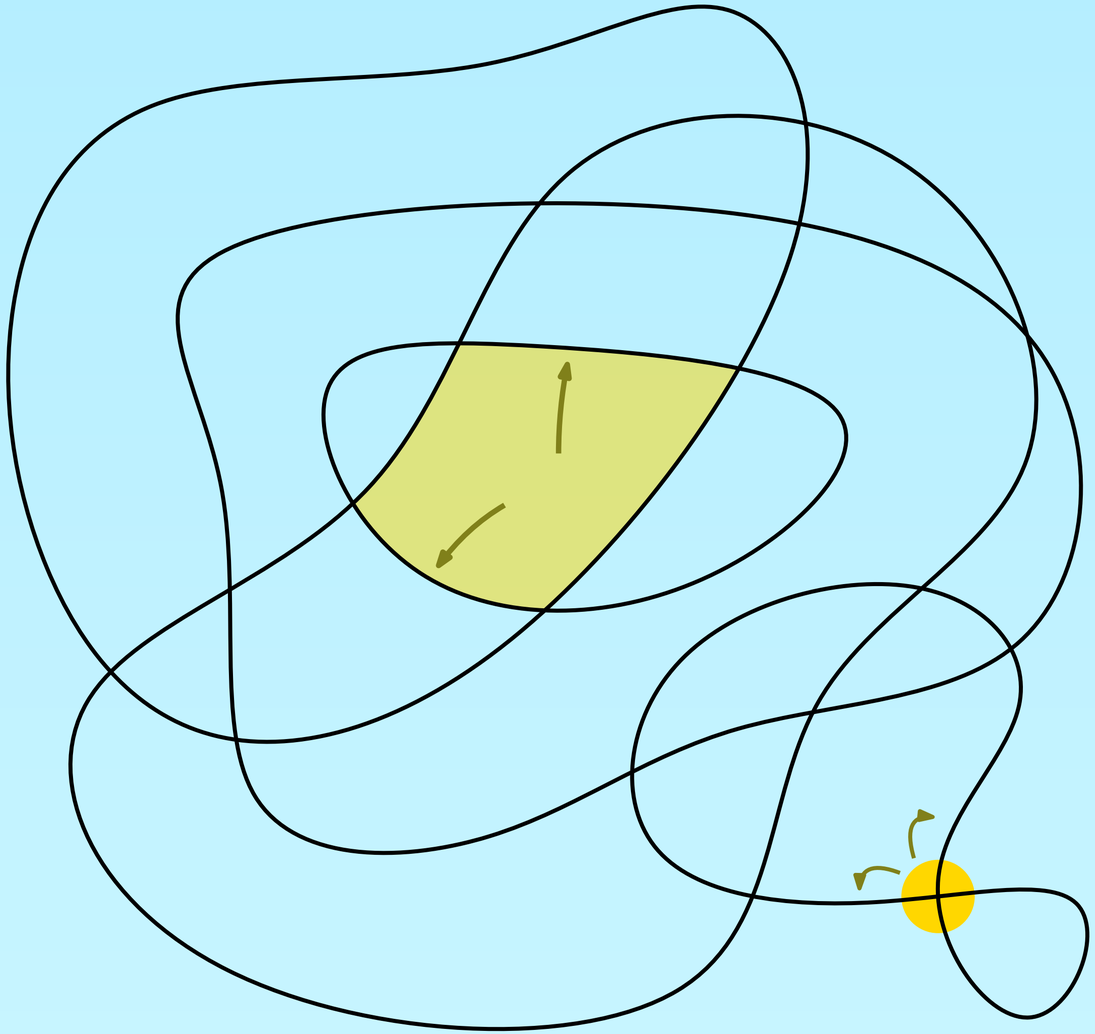
A vertex of the arrangement that is incident to the same curve twice is a self-intersection.



Consider an arrangement of closed curves in the plane.

A vertex of the arrangement that is incident to the same curve twice is a self-intersection.

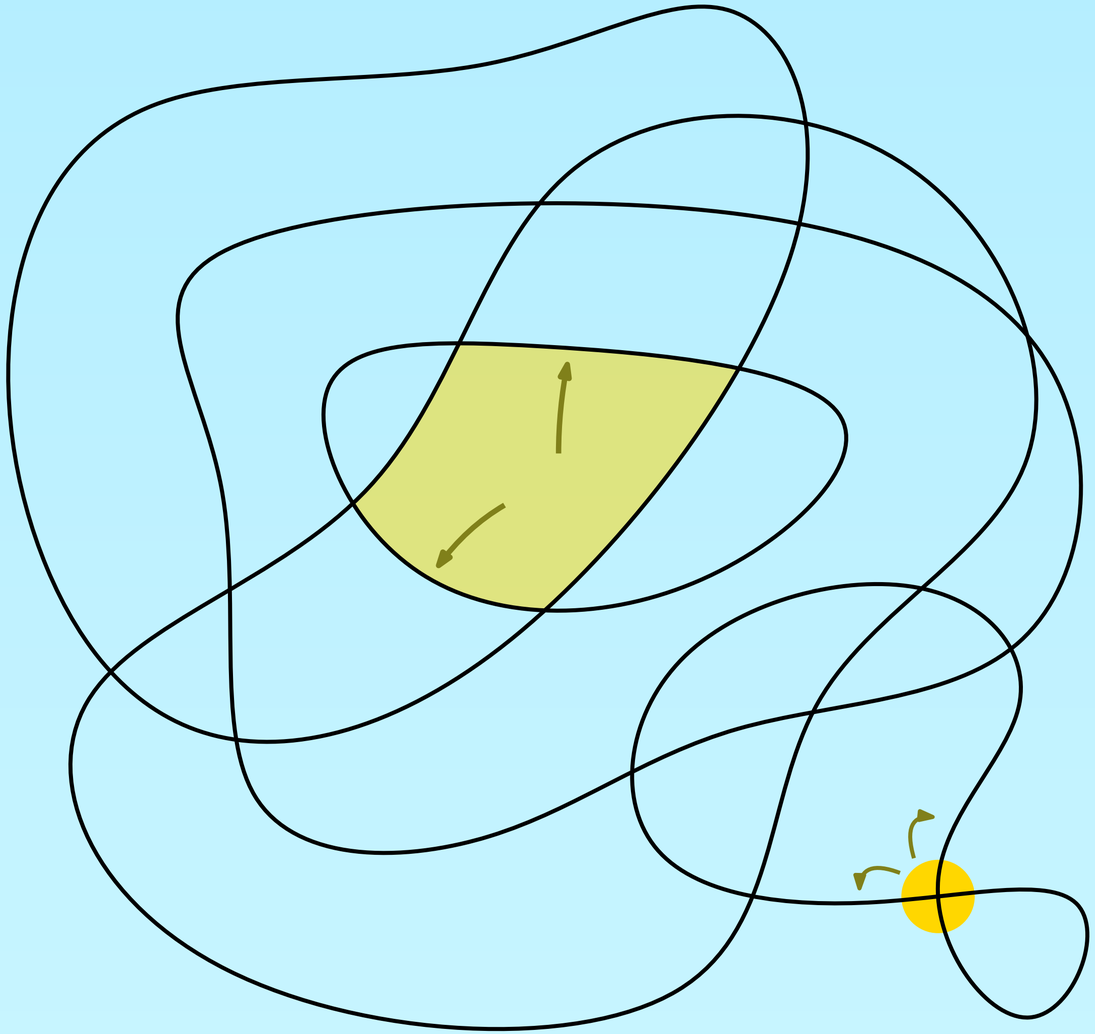
A face of the arrangement that is incident to the same curve twice is a popular face.



Consider an arrangement of closed curves in the plane.

A vertex of the arrangement that is incident to the same curve twice is a self-intersection.

A face of the arrangement that is incident to the same curve twice is a popular face.



Consider an arrangement of closed curves in the plane.

A vertex of the arrangement that is incident to the same curve twice is a self-intersection.

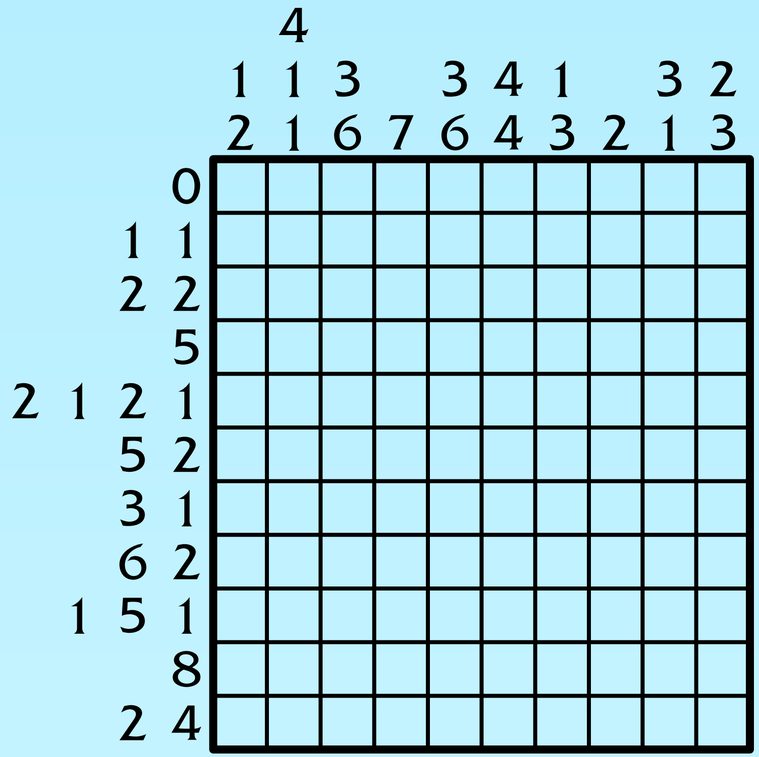
A face of the arrangement that is incident to the same curve twice is a popular face.

We want to get rid of self-intersections and popular faces.

Part 1

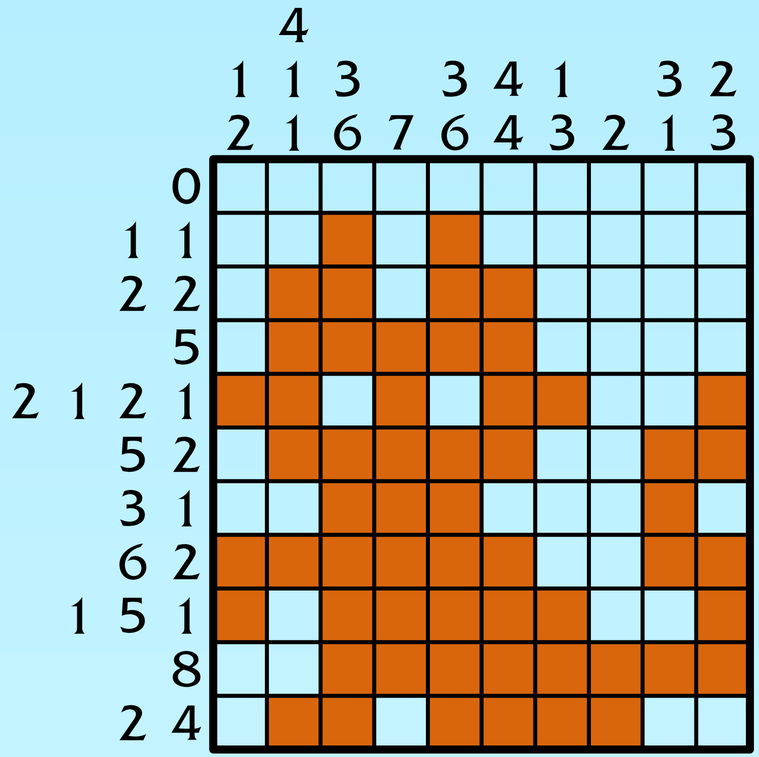
WHY?

Let's talk about Nonograms (a.k.a. Griddlers, Japanese puzzles, Hanjie, Picross, ...)



Let's talk about Nonograms (a.k.a. Griddlers, Japanese puzzles, Hanjie, Picross, ...)

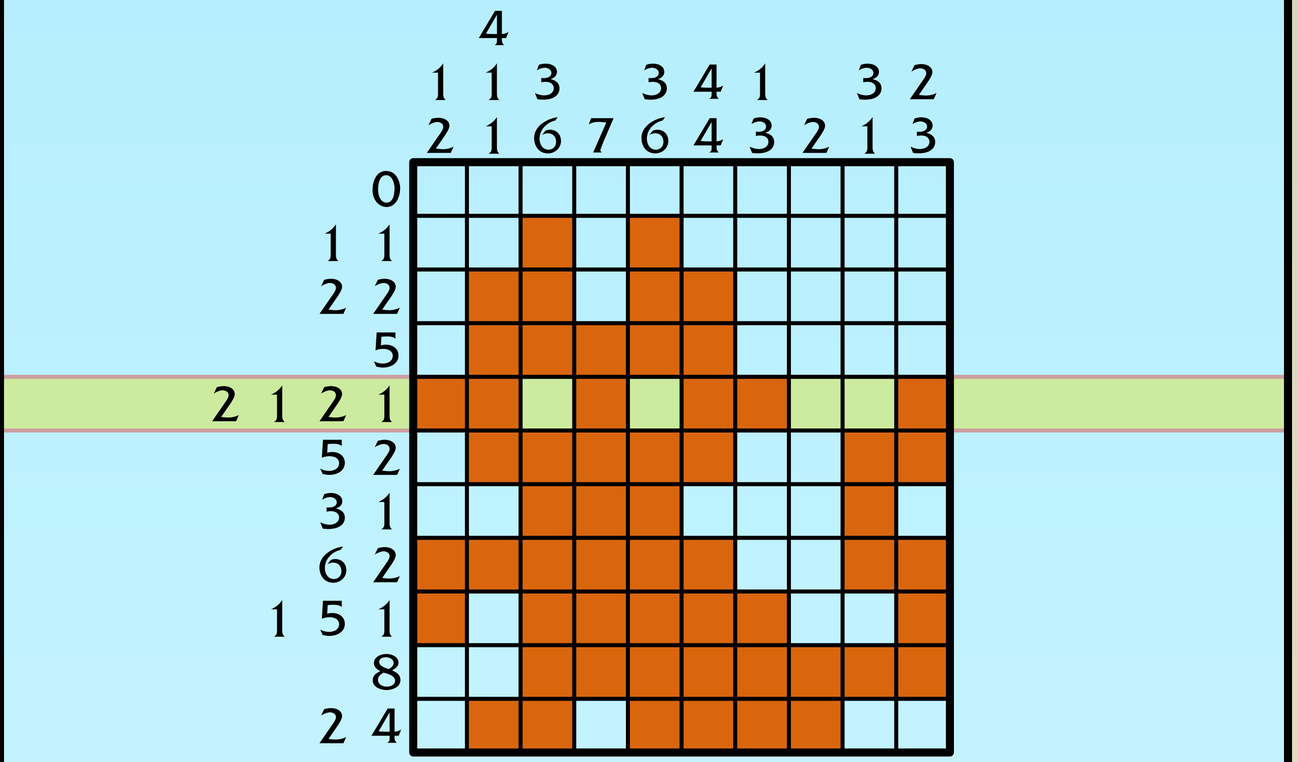
Input: a grid with some numbers next to and above it.



Let's talk about Nonograms (a.k.a. Griddlers, Japanese puzzles, Hanjie, Picross, ...)

Input: a grid with some numbers next to and above it.

Output: a black/white coloring of the grid cells.



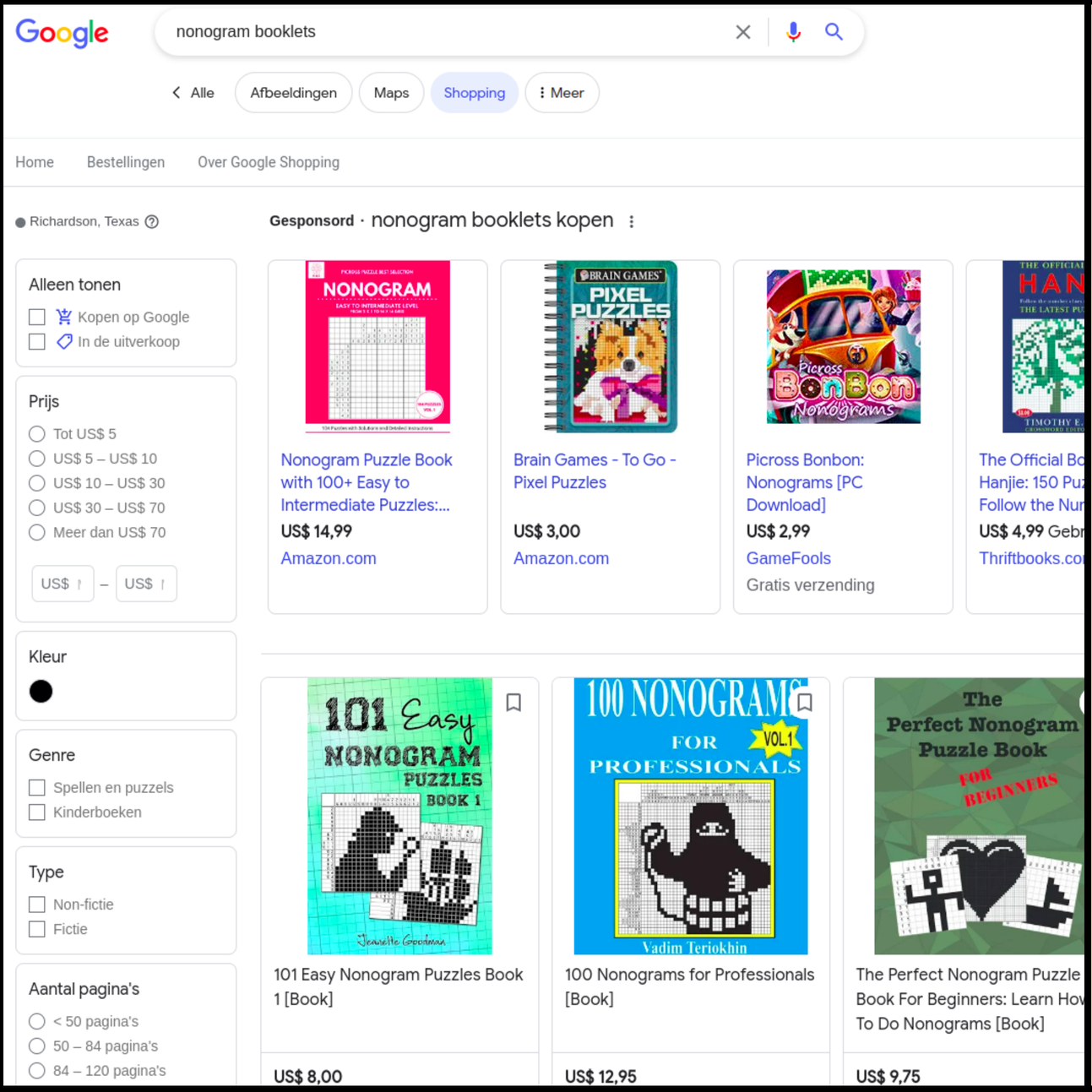
Let's talk about Nonograms (a.k.a. Griddlers, Japanese puzzles, Hanjie, Picross, ...)

Input: a grid with some numbers next to and above it.

Output: a black/white coloring of the grid cells.

Rule: in each row and column, the numbers correspond to the sizes of the consecutive blocks of black cells.

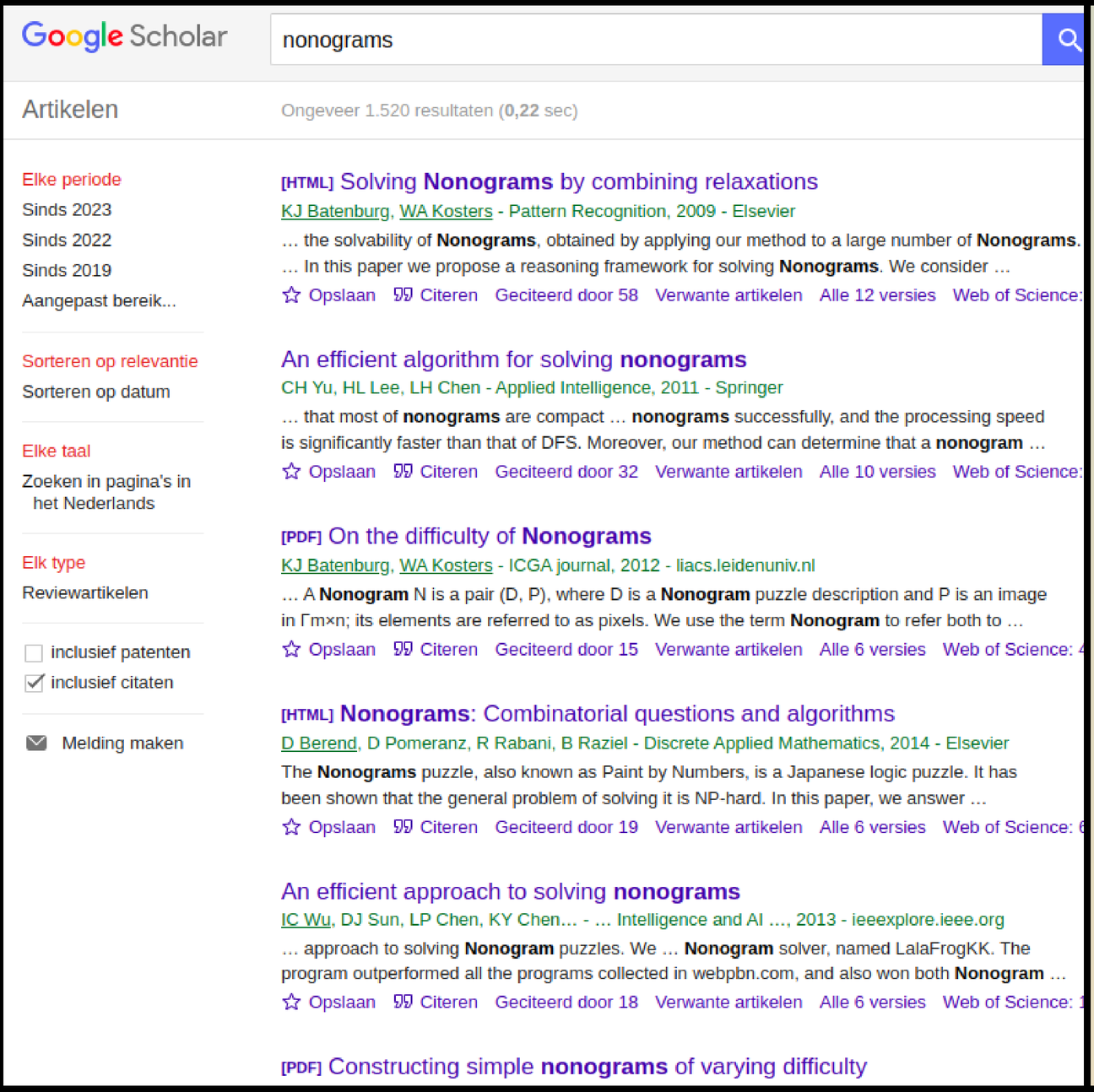
Let's talk about Nonograms (a.k.a. Griddlers, Japanese puzzles, Hanjie, Picross, ...)



Let's talk about Nonograms (a.k.a. Griddlers, Japanese puzzles, Hanjie, Picross, ...)

Very popular with consumers

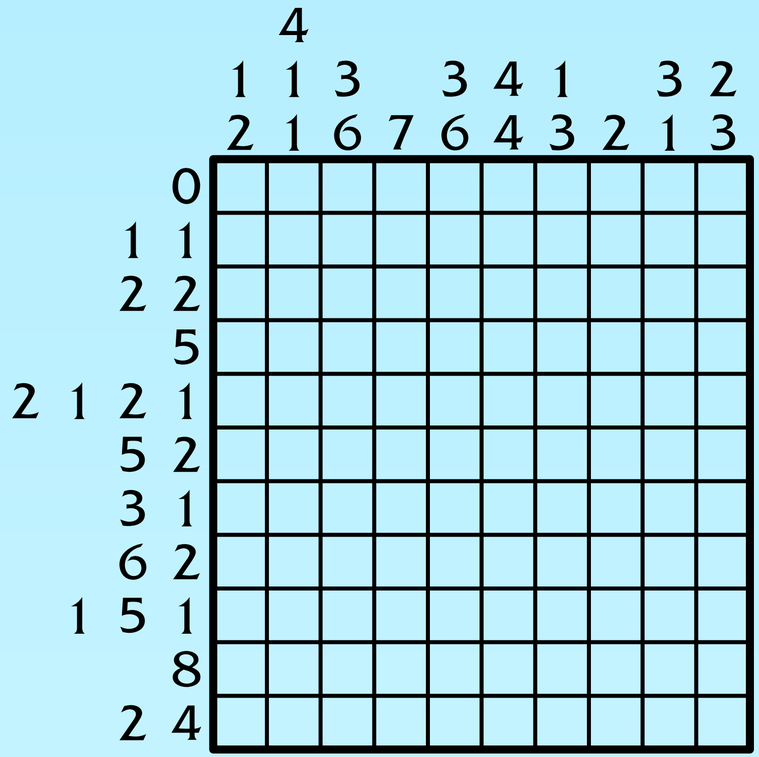
...



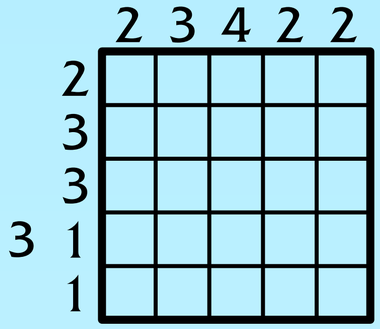
Let's talk about Nonograms (a.k.a. Griddlers, Japanese puzzles, Hanjie, Picross, ...)

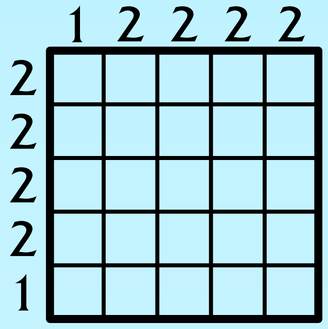
Very popular with consumers ...

... and scientists.

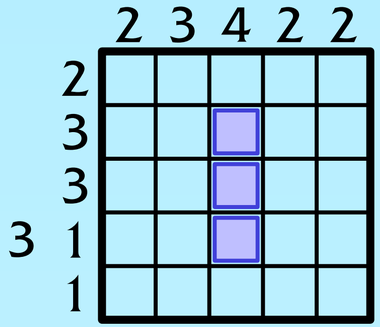


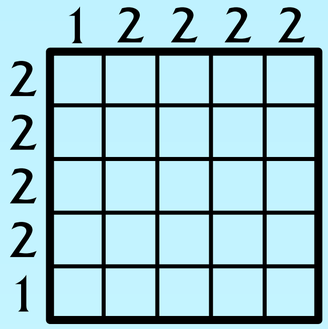
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.



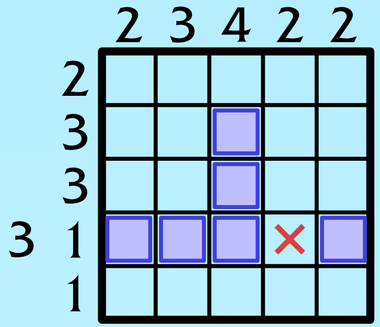


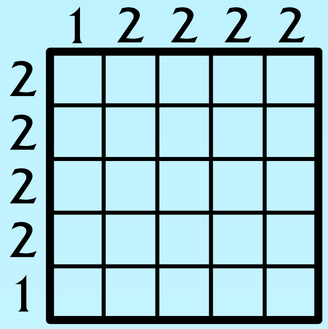
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.



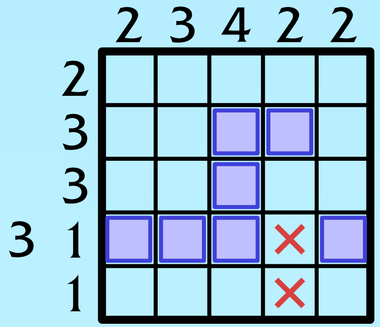


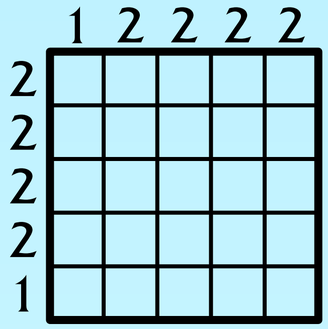
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.



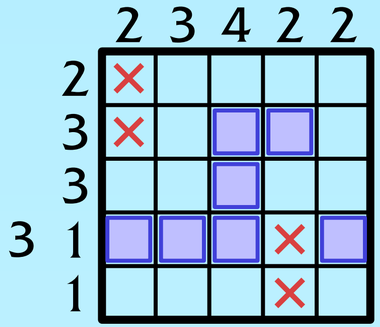


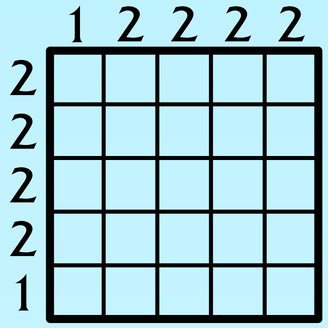
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.



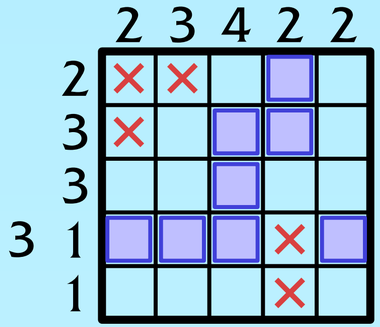


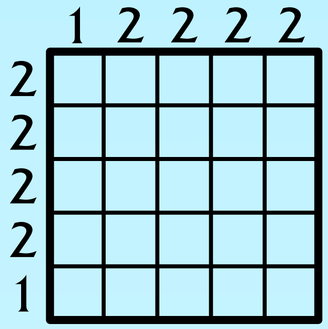
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.



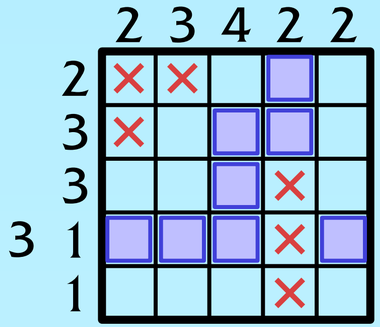


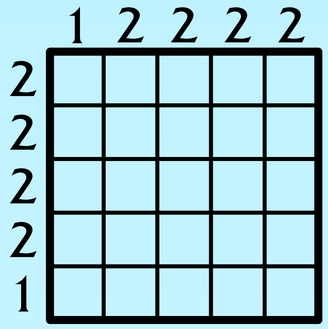
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.



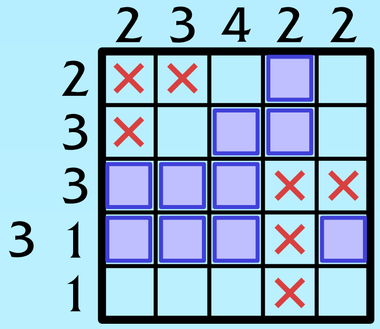


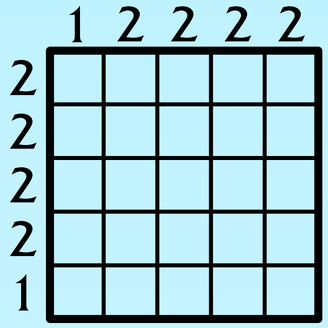
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.



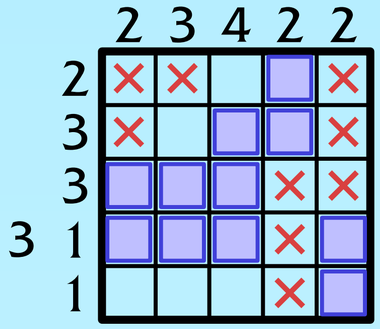


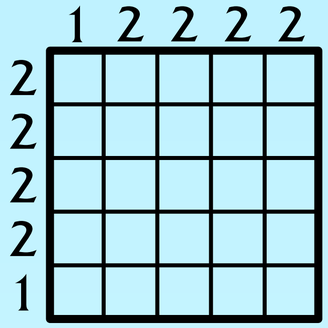
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.



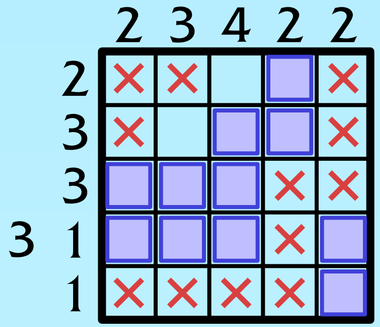


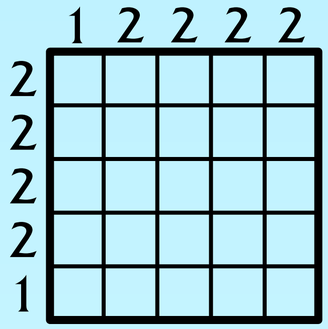
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.



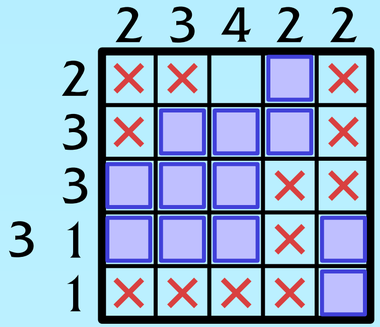


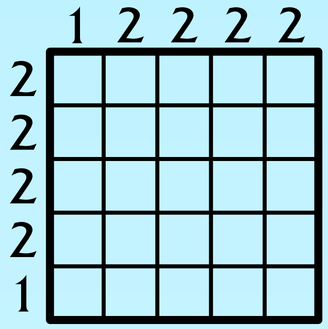
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.



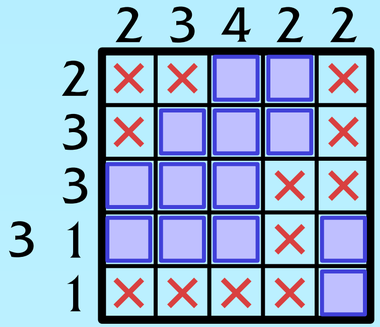


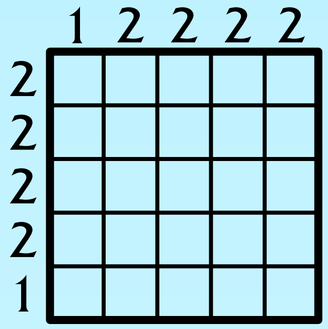
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.



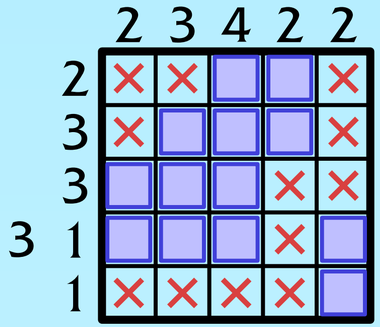


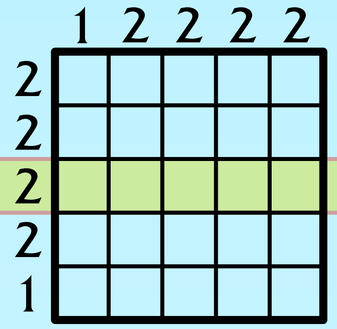
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.



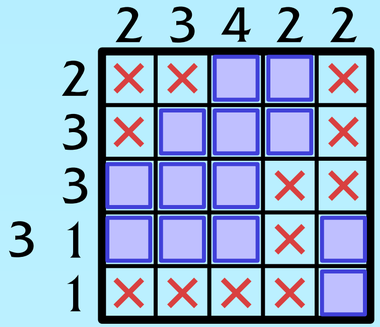


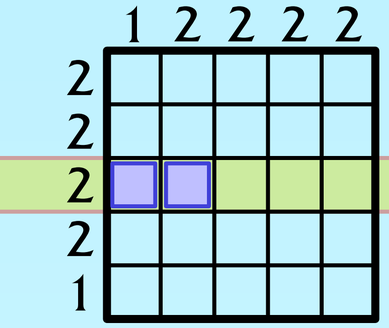
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.



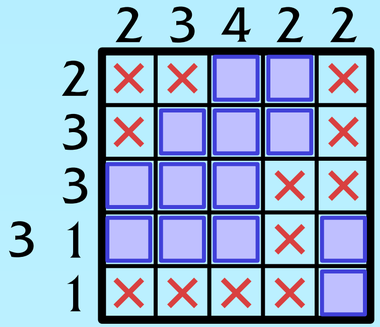


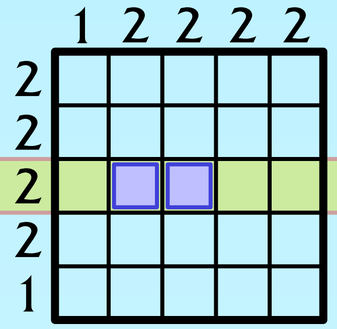
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.



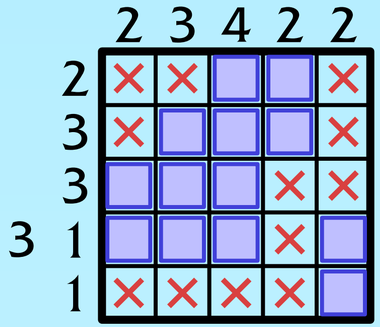


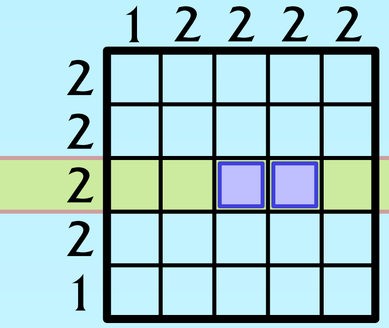
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.



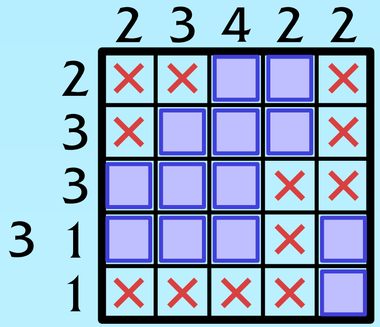


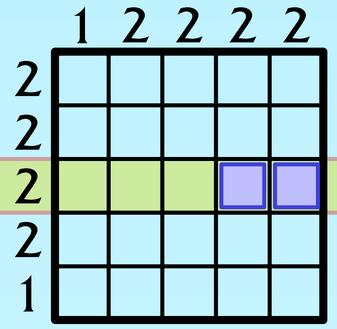
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.



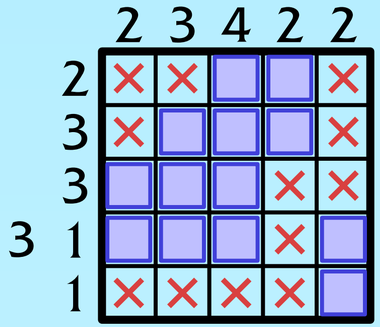


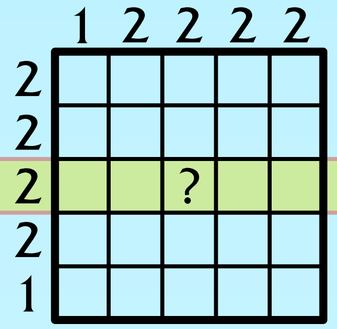
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.



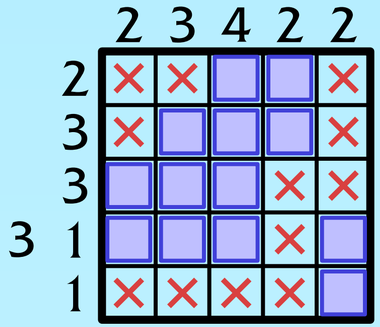


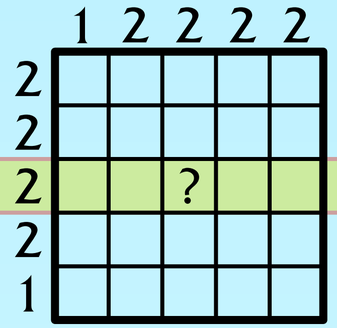
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.





A nonogram is simple if it can be solved by only ever looking at one row or column at a time.

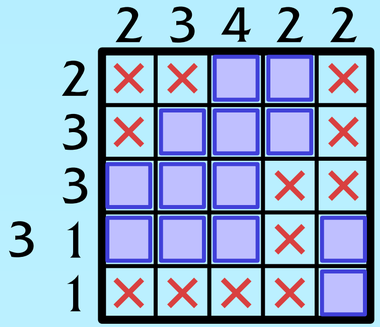


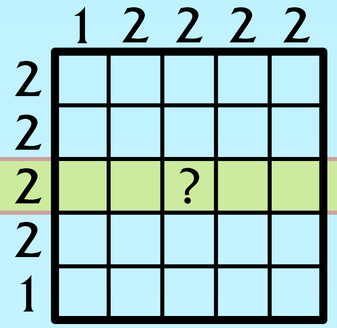


A nonogram is simple if it can be solved by only ever looking at one row or column at a time.

Solving nonograms is NP-hard, but solving simple nonograms is not.

[Batenburg & Kosters, ICGA Journal, 2012]



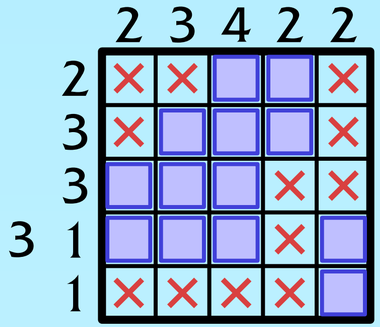


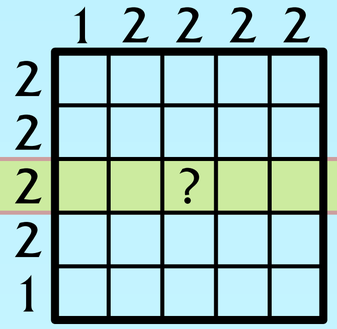
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.

Solving nonograms is NP-hard, but solving simple nonograms is not.

→ It might be of interest to generate simple nonograms.

[Batenburg & Kosters, ICGA Journal, 2012]





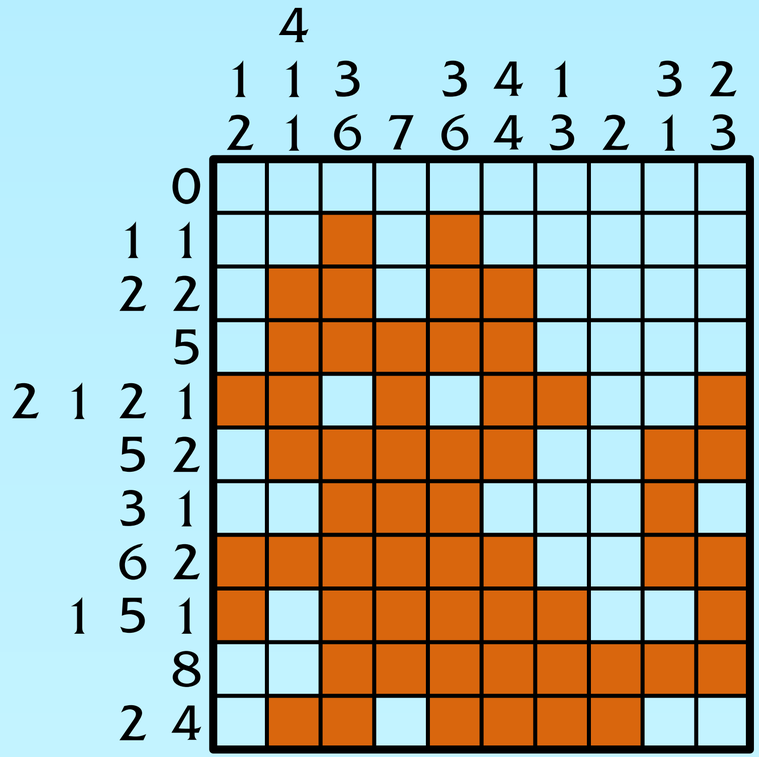
A nonogram is simple if it can be solved by only ever looking at one row or column at a time.

Solving nonograms is NP-hard, but solving simple nonograms is not.

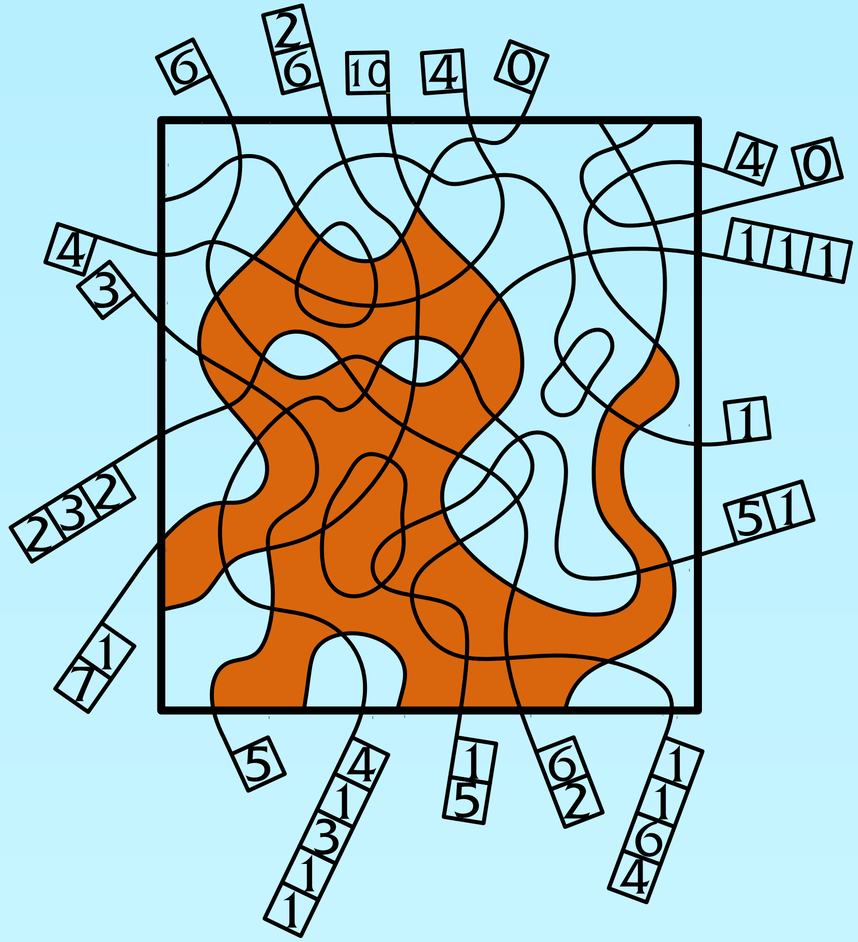
→ It might be of interest to generate simple nonograms.

More fine-grained measures of difficulty have also been studied.

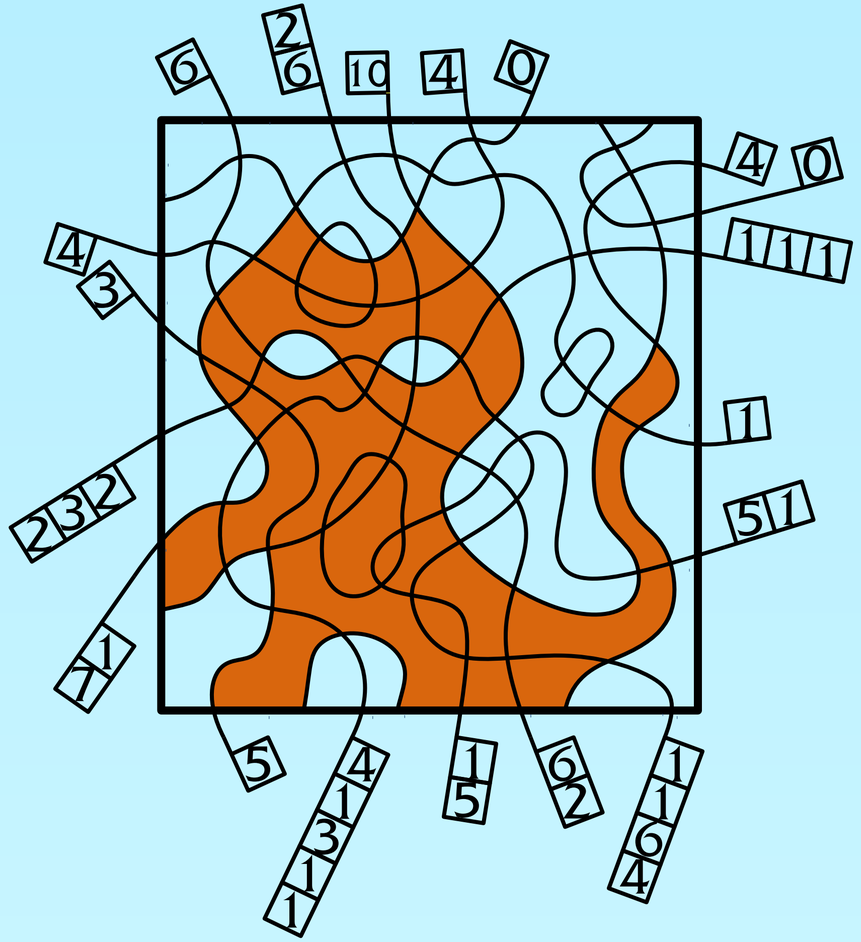
[Batenburg & Kosters, ICGA Journal, 2012]



Curved nonograms are an adaptation of nonograms that work on curve arrangements rather than grids.

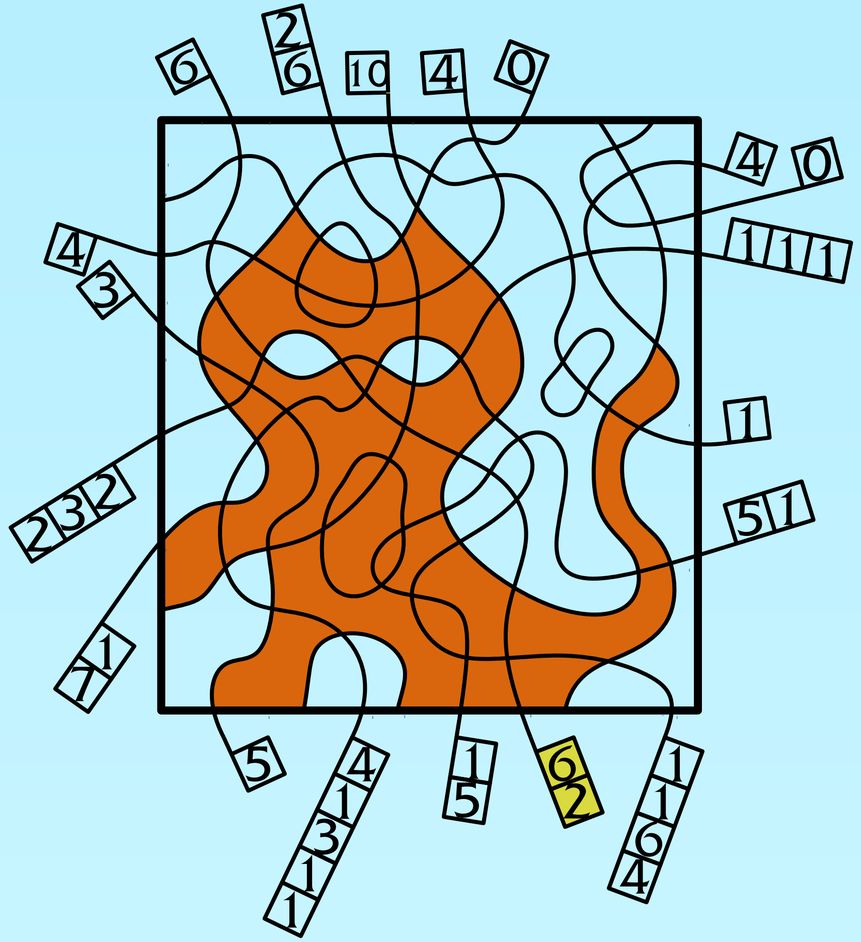


Curved nonograms are an adaptation of nonograms that work on curve arrangements rather than grids.



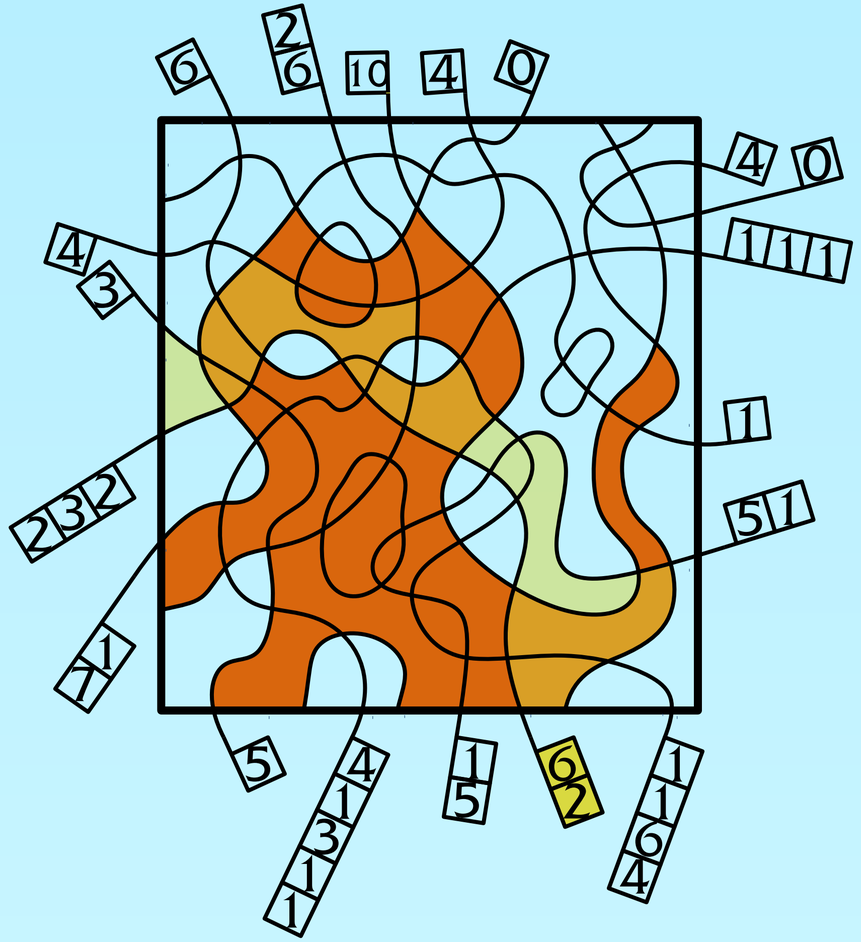
Curved nonograms are an adaptation of nonograms that work on curve arrangements rather than grids.

Similar rules as classic nonograms; clues apply to sequences of faces adjacent to a common curve.



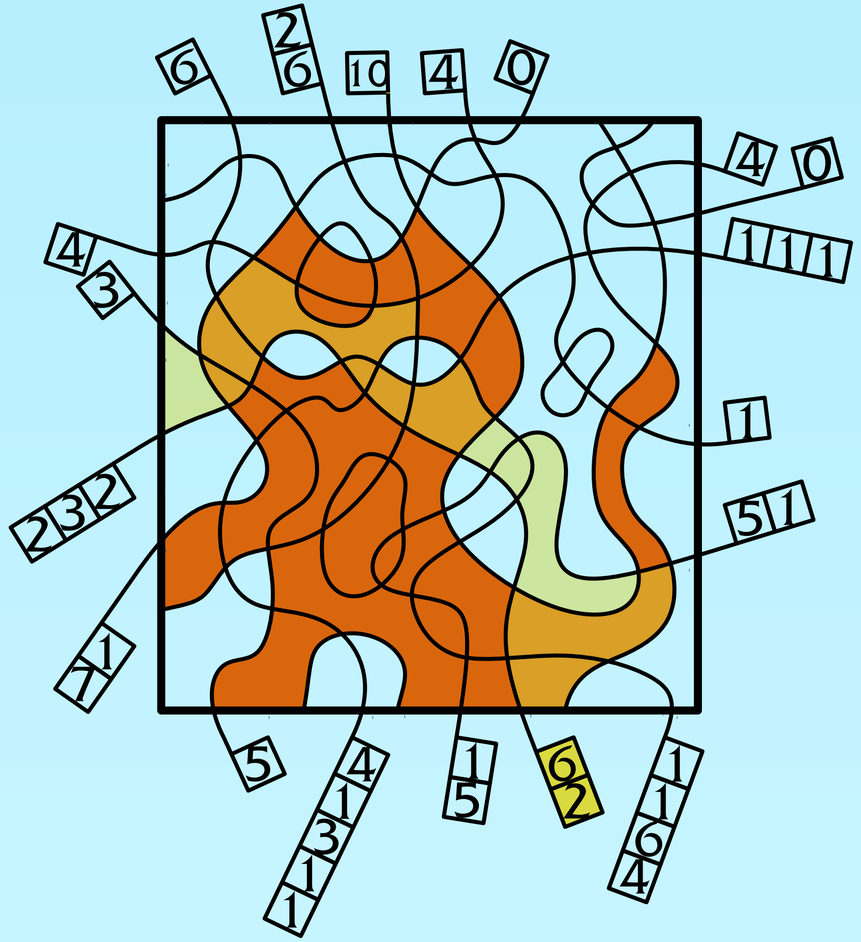
Curved nonograms are an adaptation of nonograms that work on curve arrangements rather than grids.

Similar rules as classic nonograms; clues apply to sequences of faces adjacent to a common curve.



Curved nonograms are an adaptation of nonograms that work on curve arrangements rather than grids.

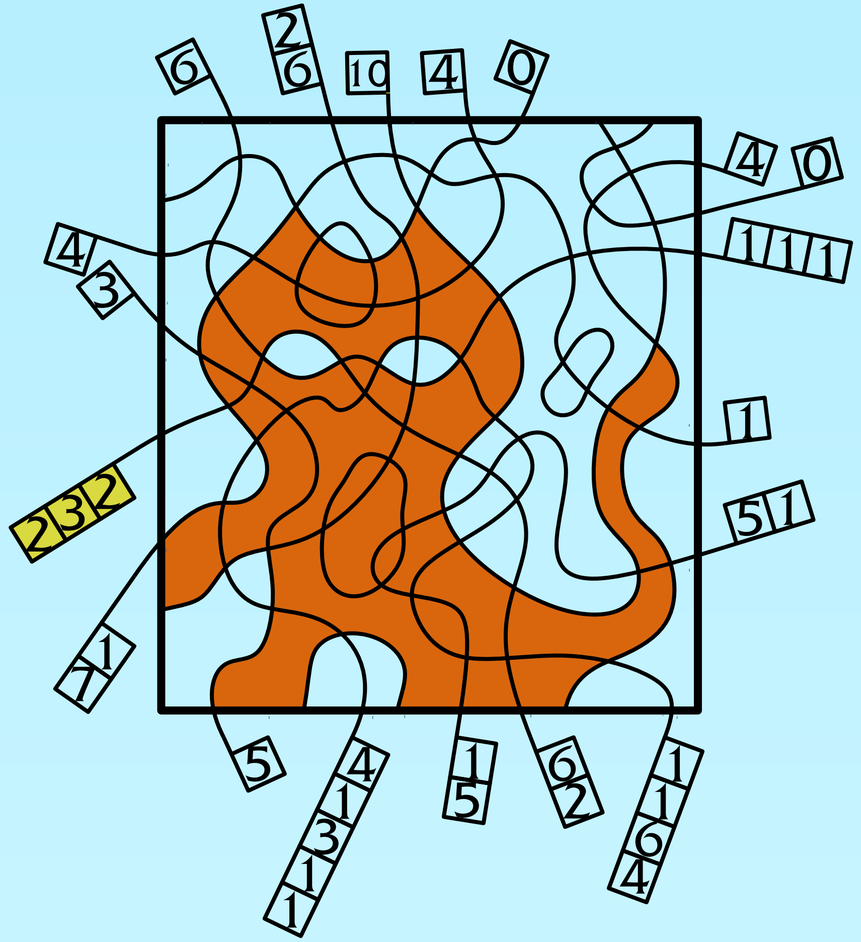
Similar rules as classic nonograms; clues apply to sequences of faces adjacent to a common curve.



Curved nonograms are an adaptation of nonograms that work on curve arrangements rather than grids.

Similar rules as classic nonograms; clues apply to sequences of faces adjacent to a common curve.

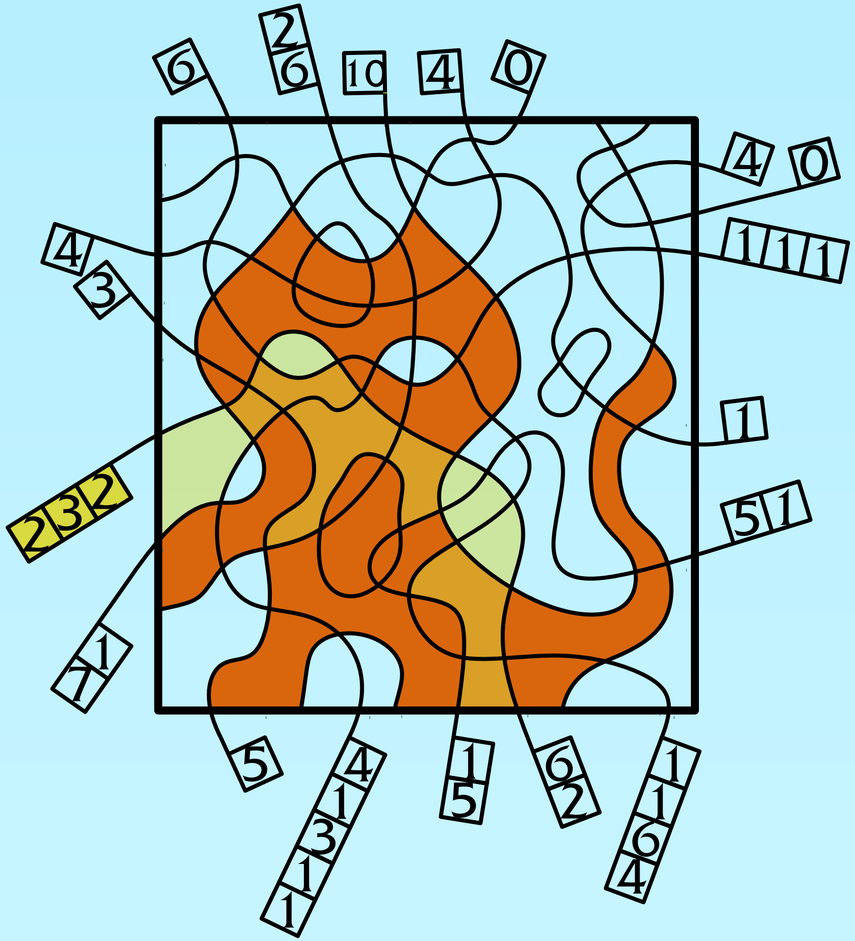
Curves have two sides with separate clues.



Curved nonograms are an adaptation of nonograms that work on curve arrangements rather than grids.

Similar rules as classic nonograms; clues apply to sequences of faces adjacent to a common curve.

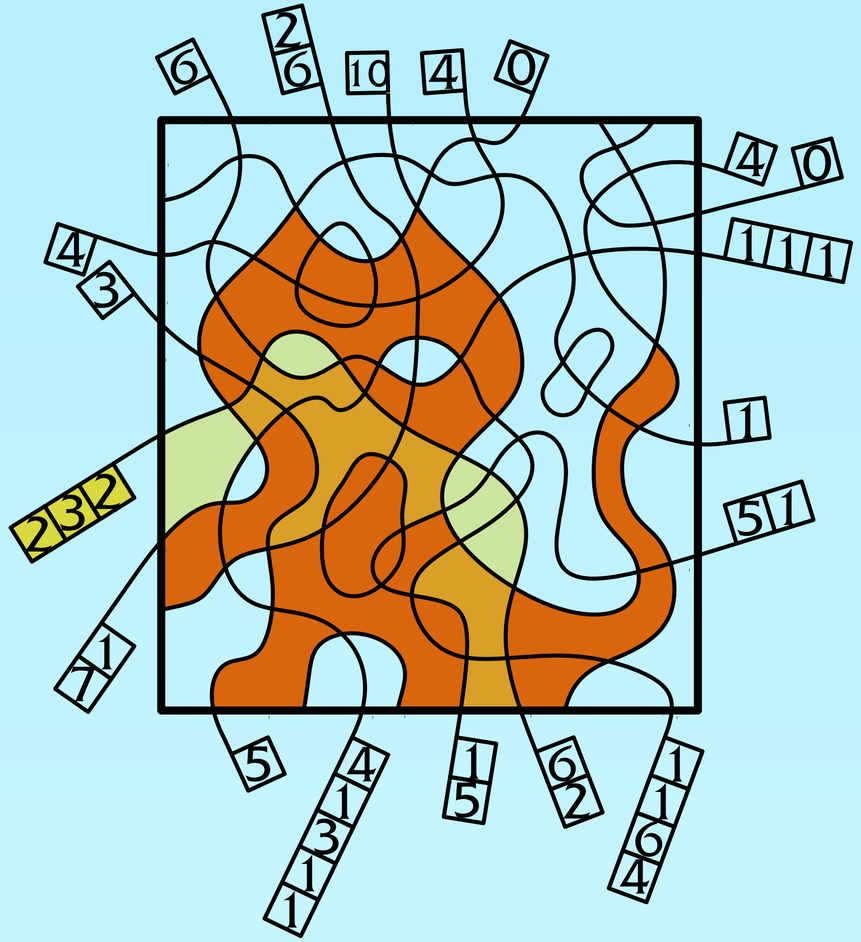
Curves have two sides with separate clues.



Curved nonograms are an adaptation of nonograms that work on curve arrangements rather than grids.

Similar rules as classic nonograms; clues apply to sequences of faces adjacent to a common curve.

Curves have two sides with separate clues.

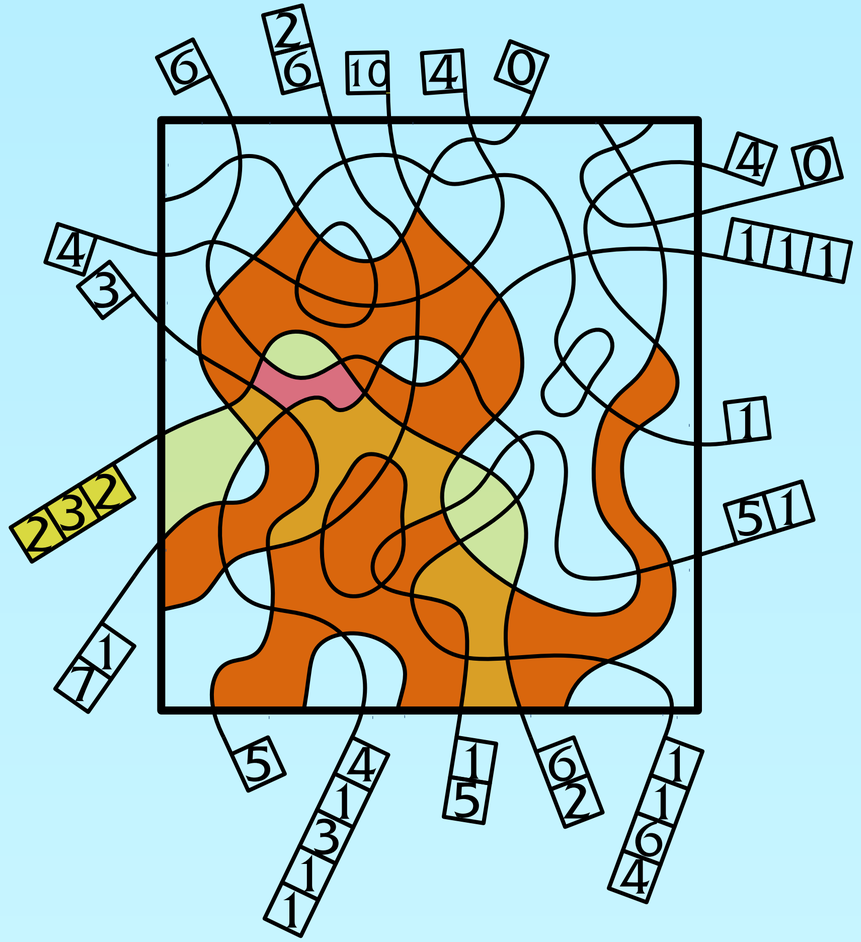


Curved nonograms are an adaptation of nonograms that work on curve arrangements rather than grids.

Similar rules as classic nonograms; clues apply to sequences of faces adjacent to a common curve.

Curves have two sides with separate clues.

Cells can appear twice!



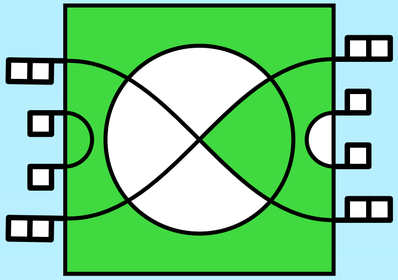
Curved nonograms are an adaptation of nonograms that work on curve arrangements rather than grids.

Similar rules as classic nonograms; clues apply to sequences of faces adjacent to a common curve.

Curves have two sides with separate clues.

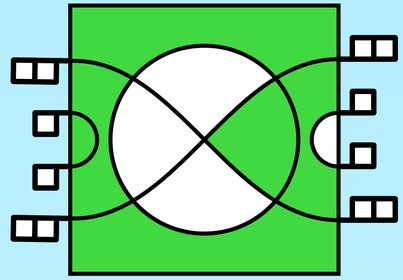
Cells can appear twice!

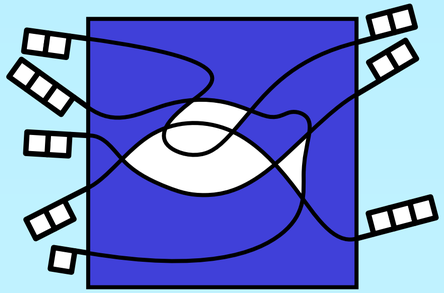
We can identify three levels of “conceptual difficulty”.



We can identify three levels of “conceptual difficulty”.

Basic nonograms do not have any duplicate cells adjacent to the same curve.

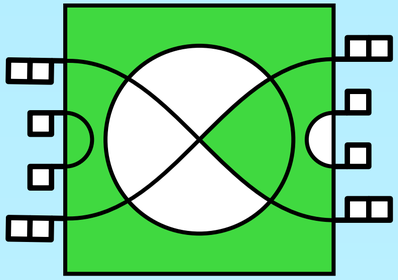


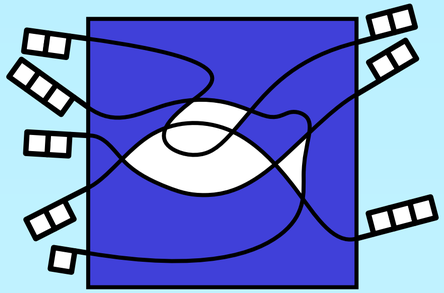


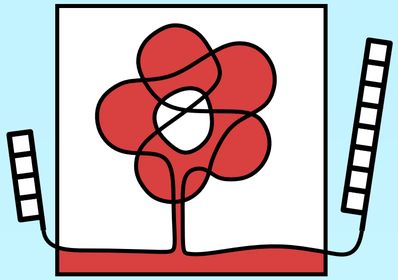
We can identify three levels of “conceptual difficulty”.

Basic nonograms do not have any duplicate cells adjacent to the same curve.

Advanced nonograms may have duplicate cells adjacent to the same side of a curve.





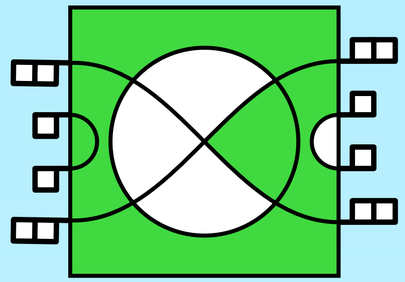


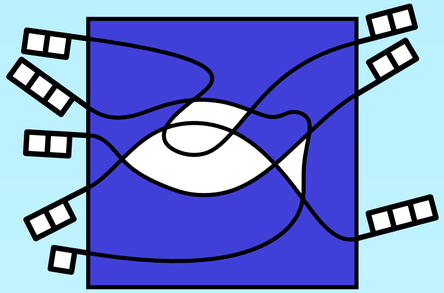
We can identify three levels of “conceptual difficulty”.

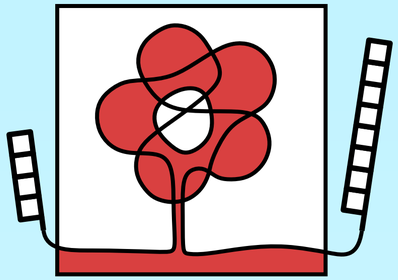
Basic nonograms do not have any duplicate cells adjacent to the same curve.

Advanced nonograms may have duplicate cells adjacent to the same side of a curve.

Expert nonograms have duplicate cells adjacent to opposite sides of a curve.

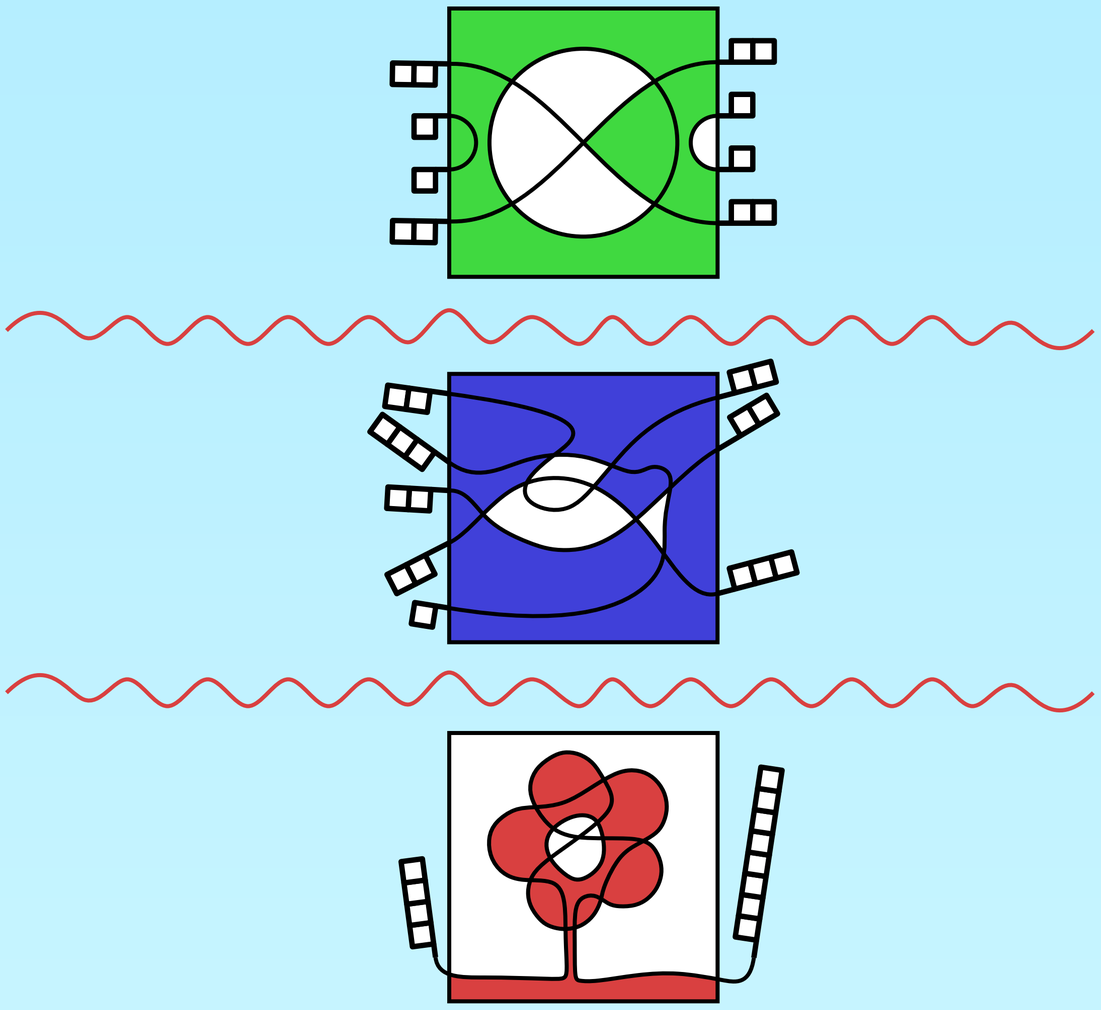






We can identify three levels of “conceptual difficulty”.

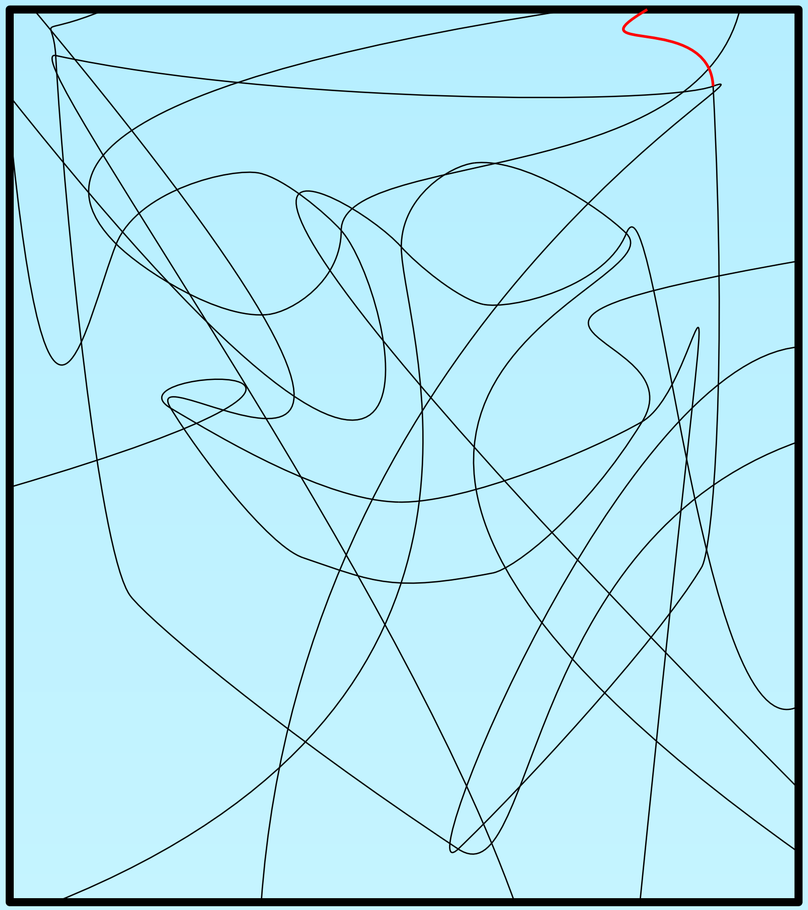
Advanced versus Expert:
presence of self-intersections



We can identify three levels of “conceptual difficulty”.

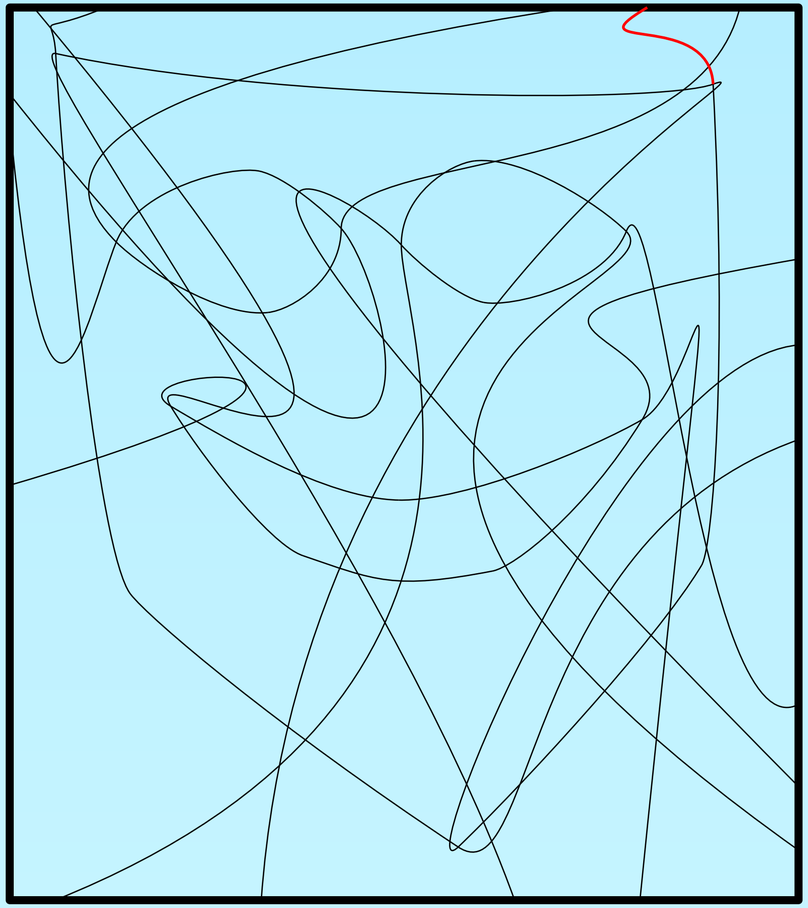
Basic versus Advanced:
presence of popular faces

Advanced versus Expert:
presence of self-intersections



One can generate nonograms from desired output pictures.

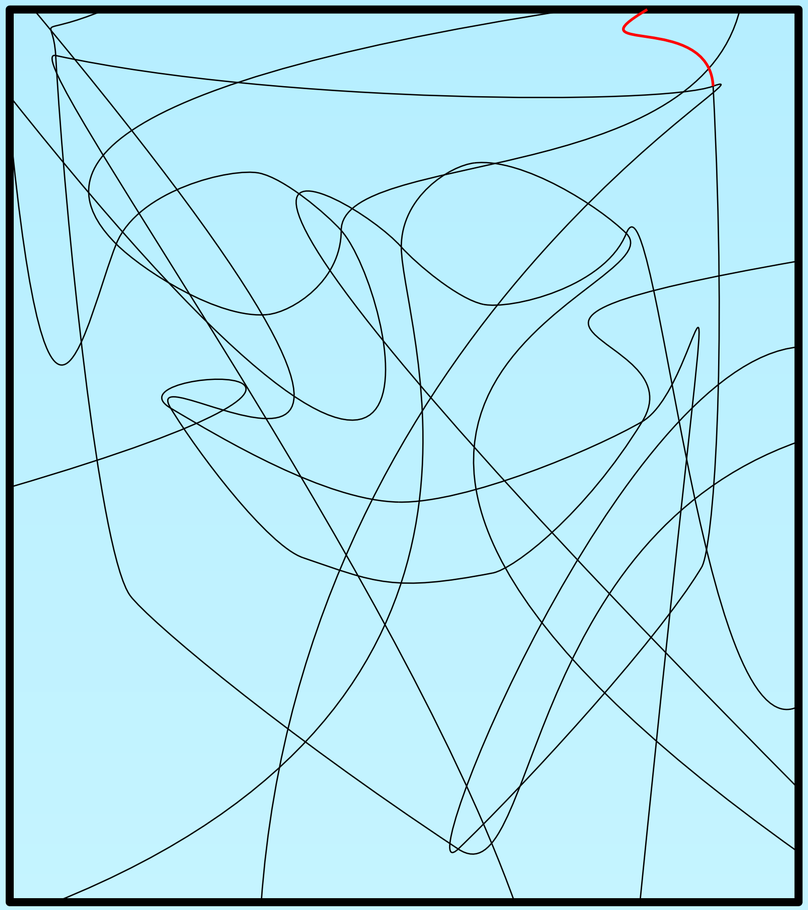
[van de Kerkhof, de Jong, Parment, L, Vaxman & van Kreveld, EuroGraphics 2019]



One can generate nonograms from desired output pictures.

Generated puzzles are often advanced or expert.

[van de Kerkhof, de Jong, Parment, L, Vaxman & van Kreveld, EuroGraphics 2019]

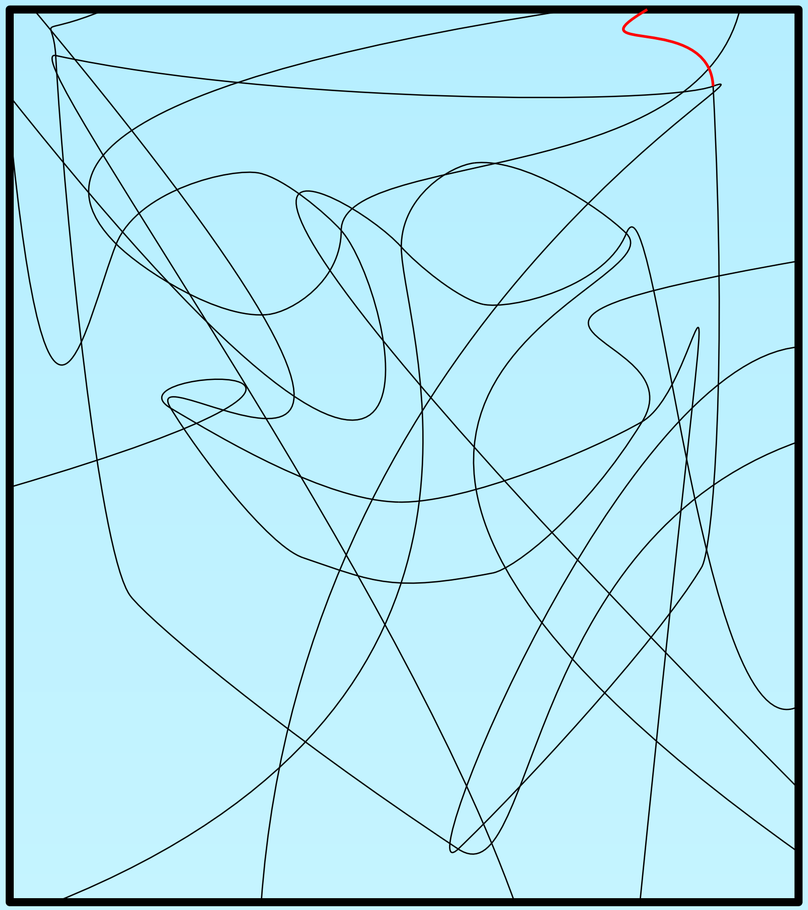


One can generate nonograms from desired output pictures.

Generated puzzles are often advanced or expert.

User studies suggest a higher demand for basic puzzles.

[van de Kerkhof, de Jong, Parment, L, Vaxman & van Kreveld, EuroGraphics 2019]



One can generate nonograms from desired output pictures.

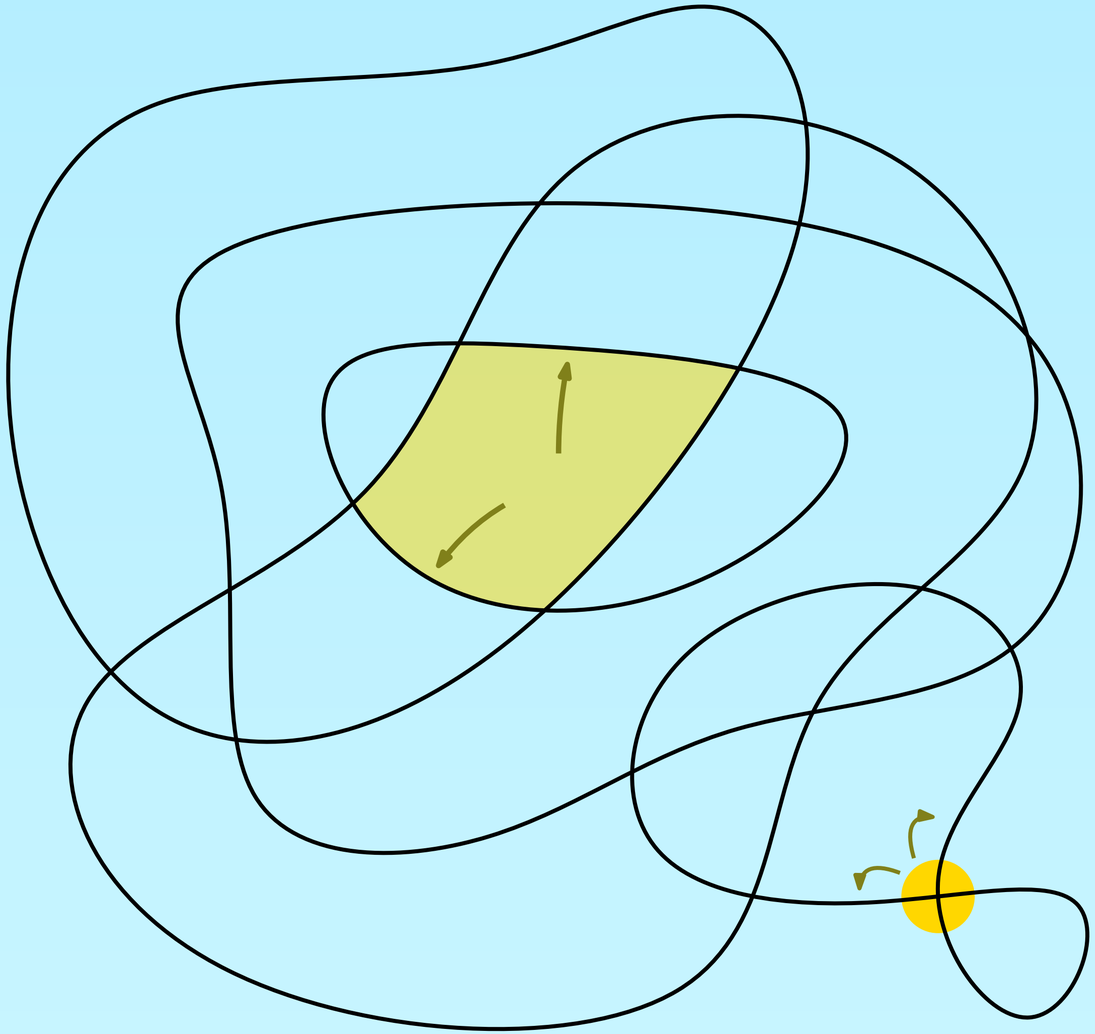
Generated puzzles are often advanced or expert.

User studies suggest a higher demand for basic puzzles.

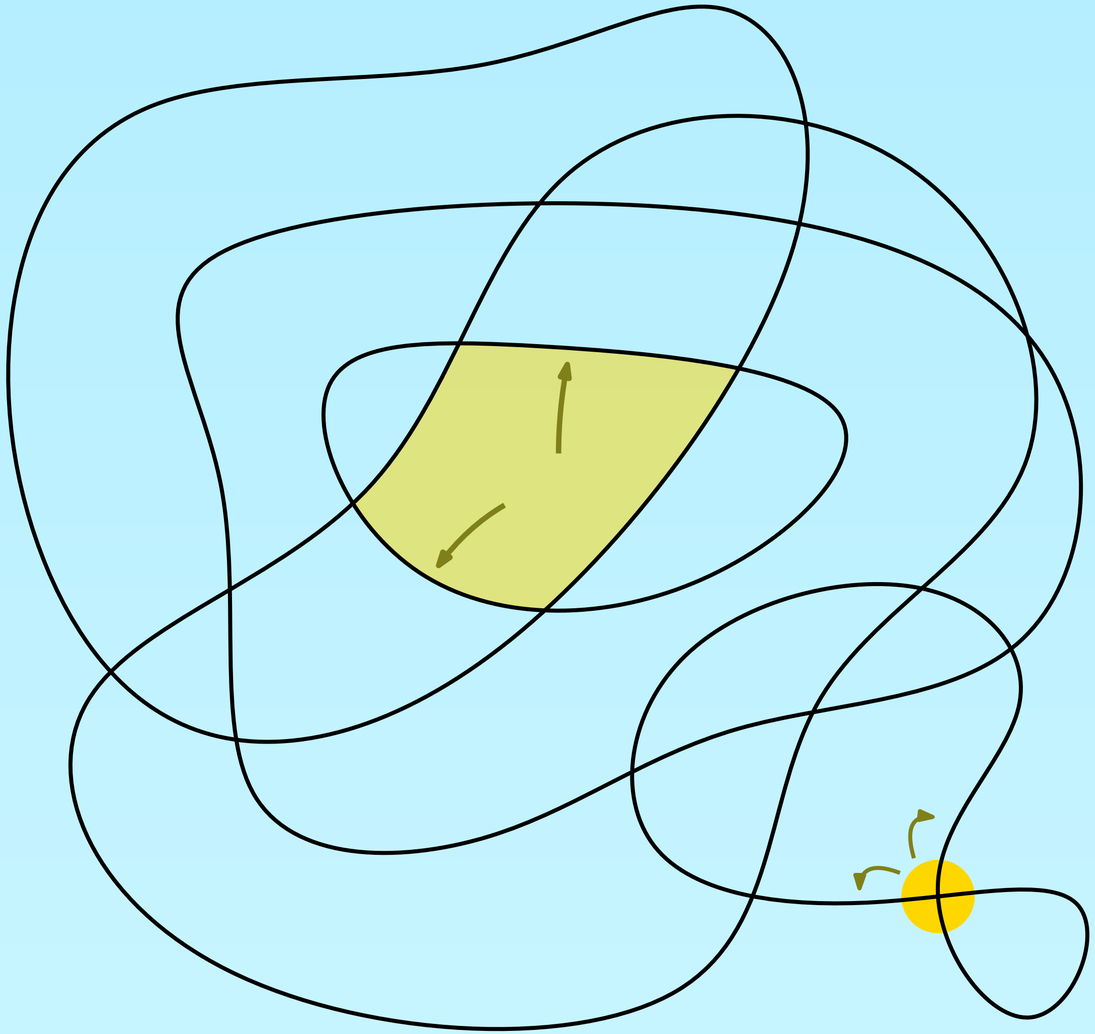
Can we turn expert puzzles into advanced or basic ones?

[van de Kerkhof, de Jong, Parment, L, Vaxman & van Kreveld, EuroGraphics 2019]

Part 2
WHAT?

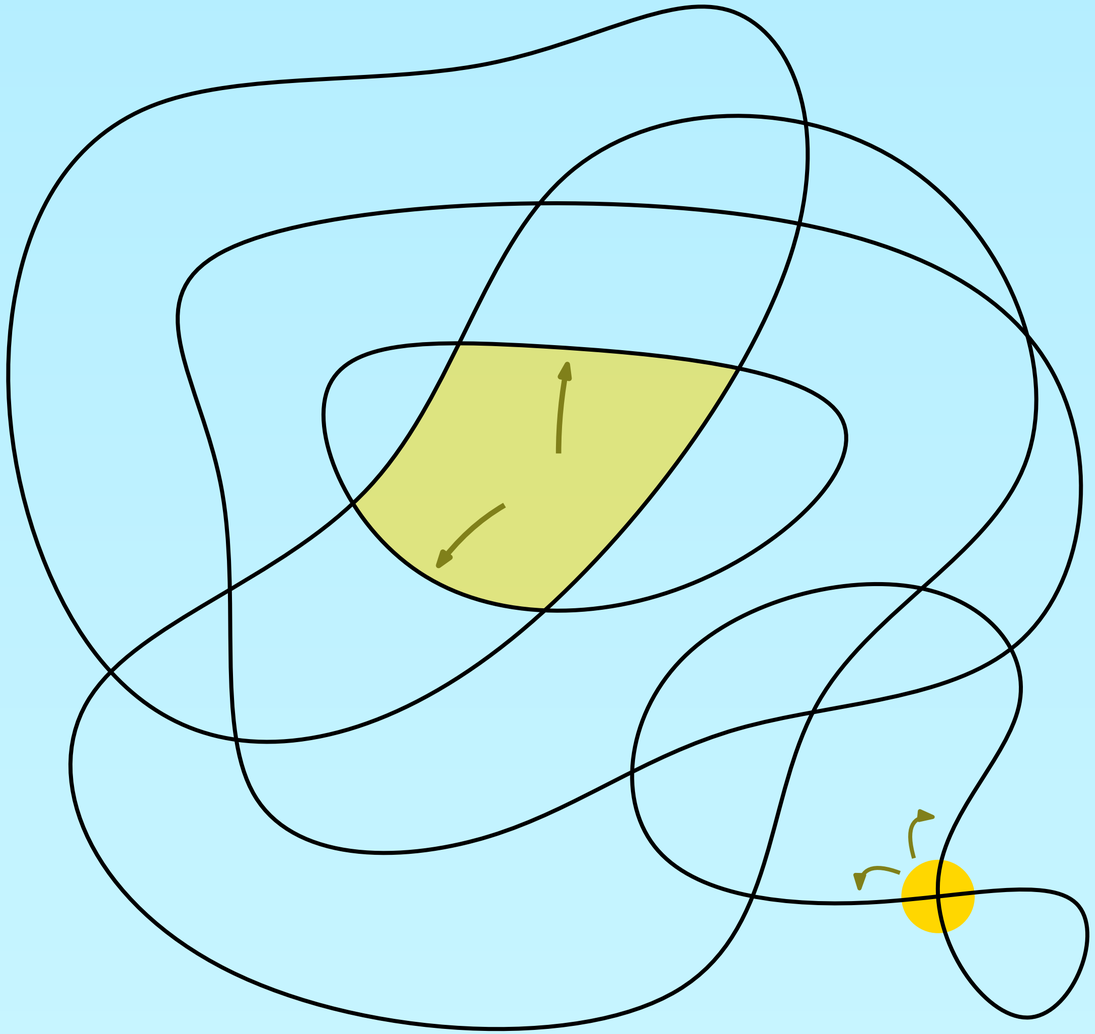


So we want to get rid of self-intersections and/or popular faces in curve arrangements.



So we want to get rid of self-intersections and/or popular faces in curve arrangements.

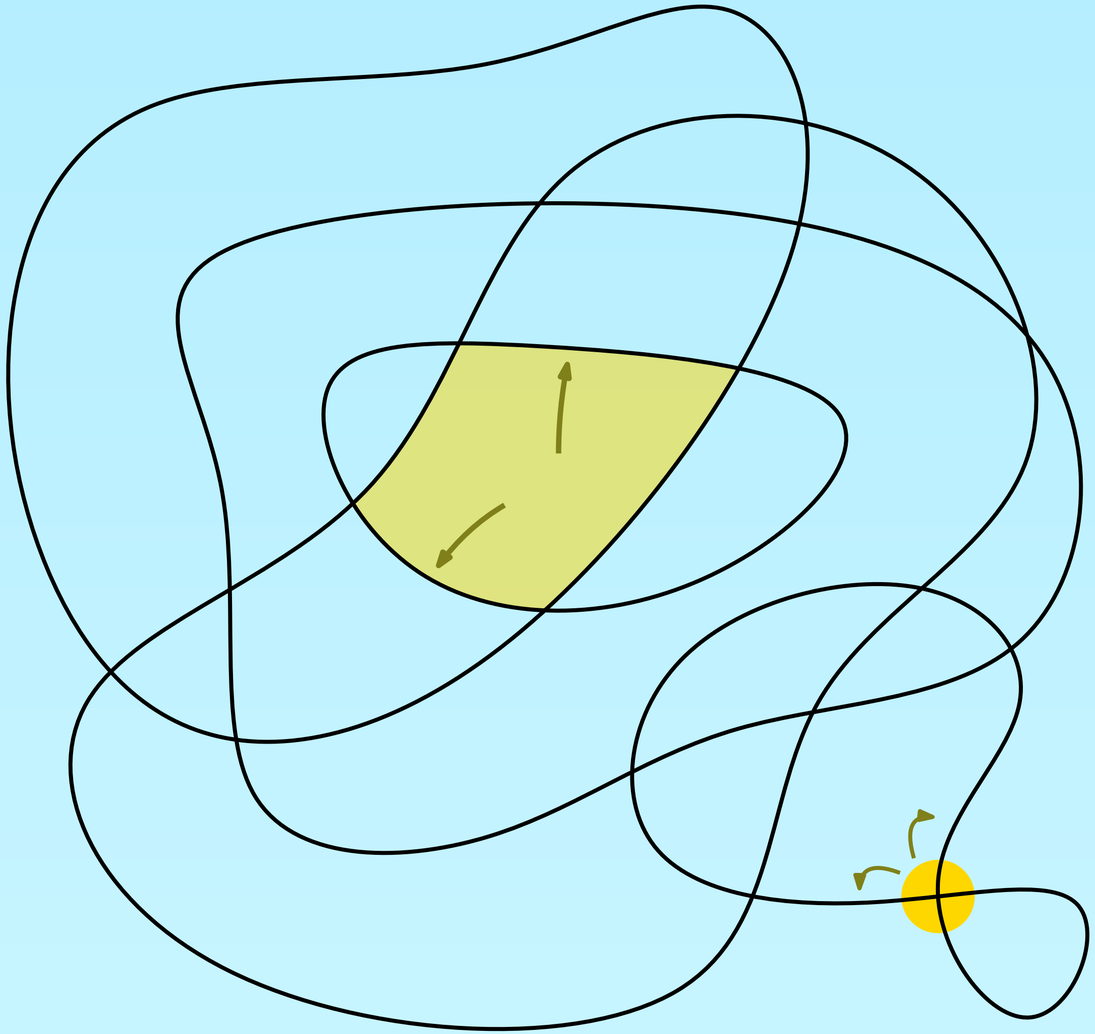
Observation. At least one face incident to every self-intersection is popular.



So we want to get rid of self-intersections and/or popular faces in curve arrangements.

Observation. At least one face incident to every self-intersection is popular.

So, we can't remove all popular faces without also removing all self-intersections.

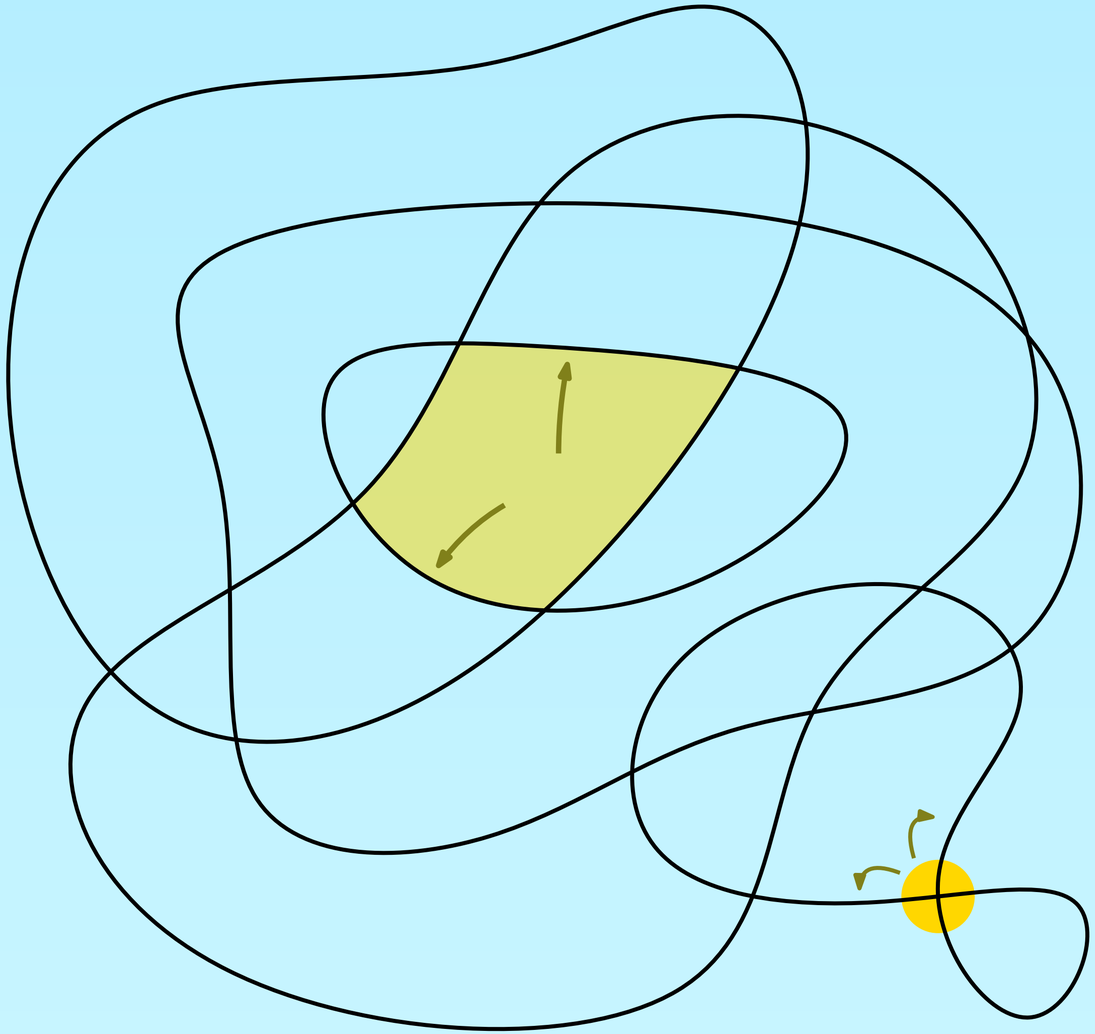


So we want to get rid of self-intersections and/or popular faces in curve arrangements.

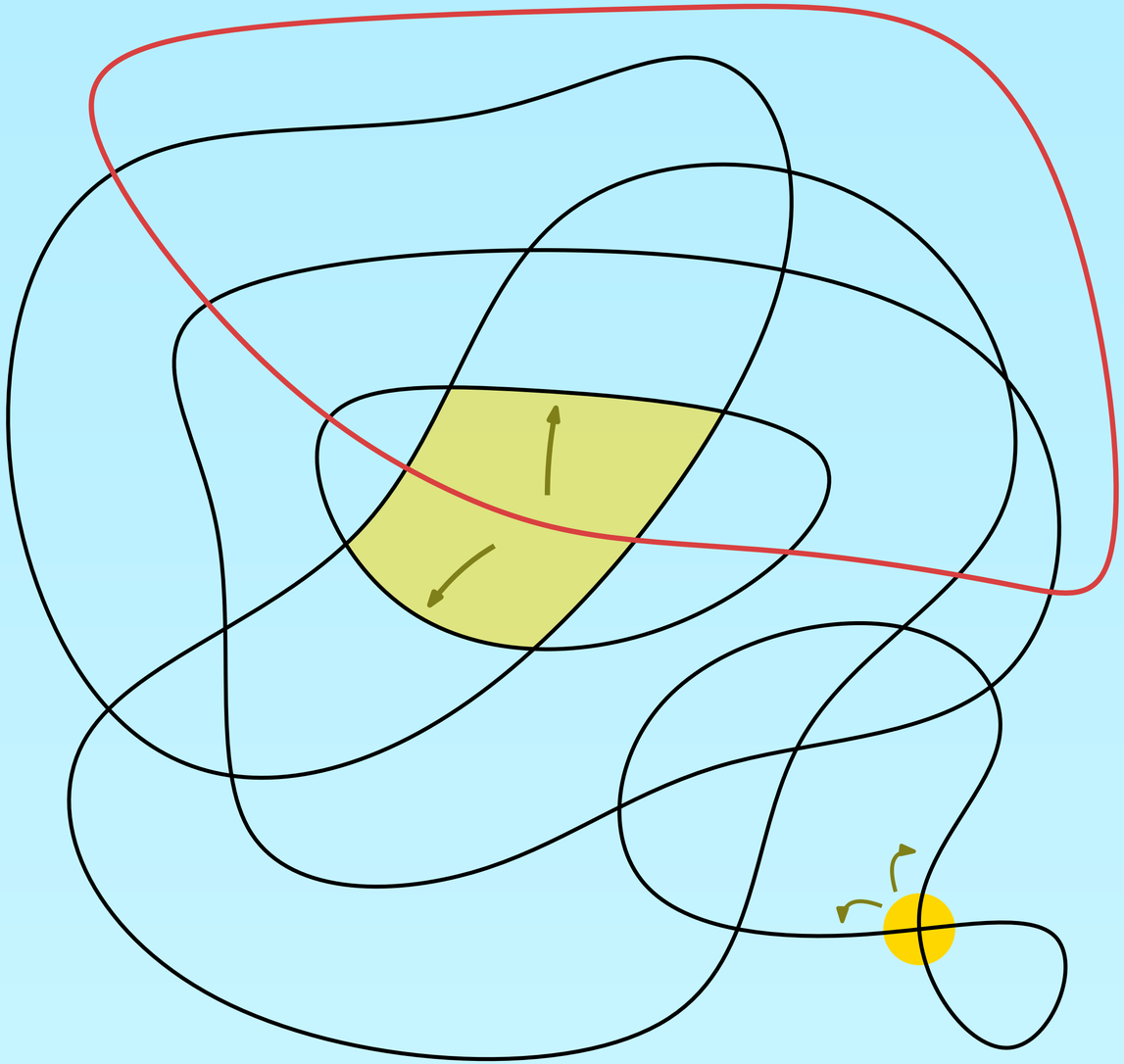
Observation. At least one face incident to every self-intersection is popular.

So, we can't remove all popular faces without also removing all self-intersections.

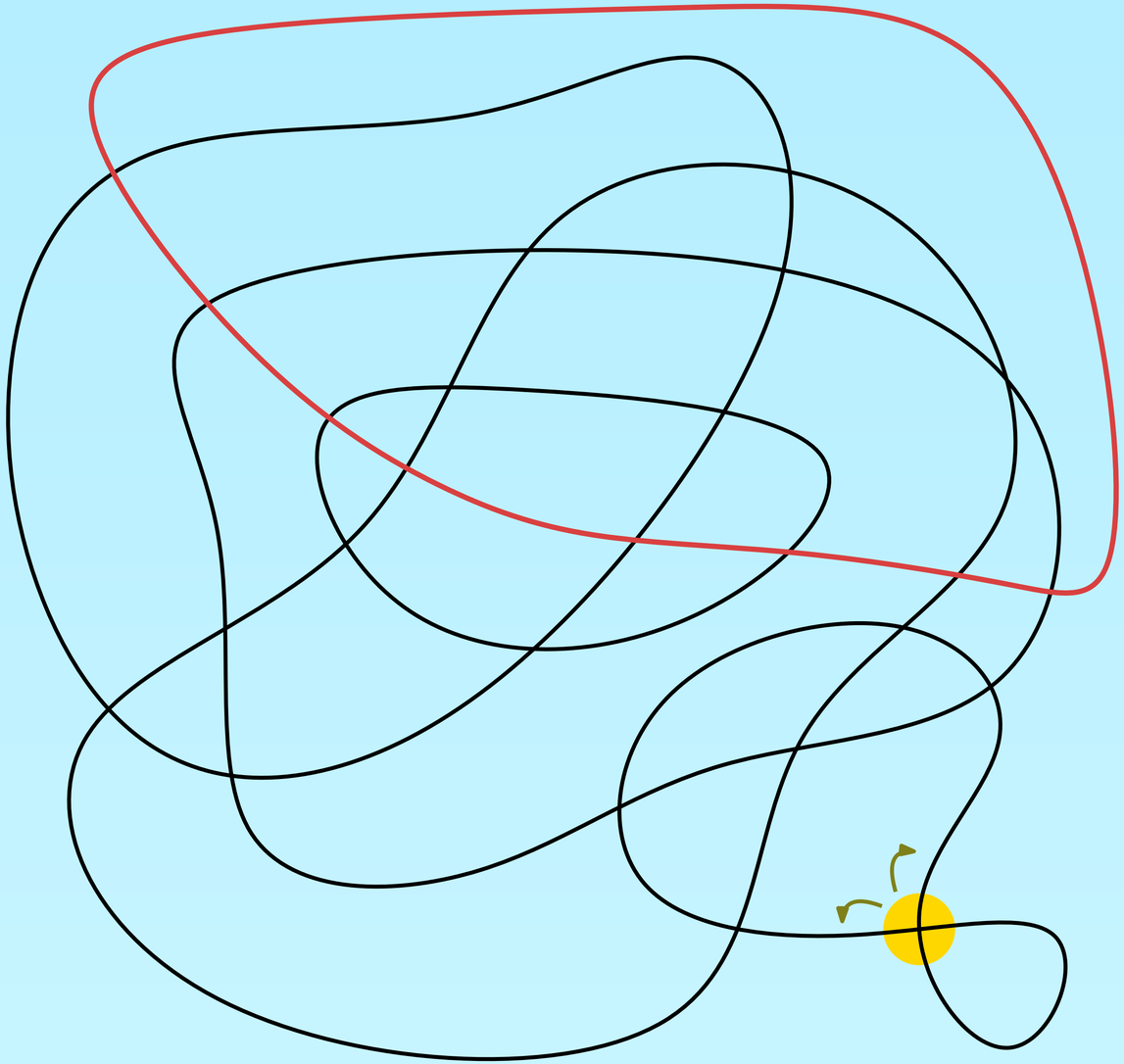
But how do we remove them?



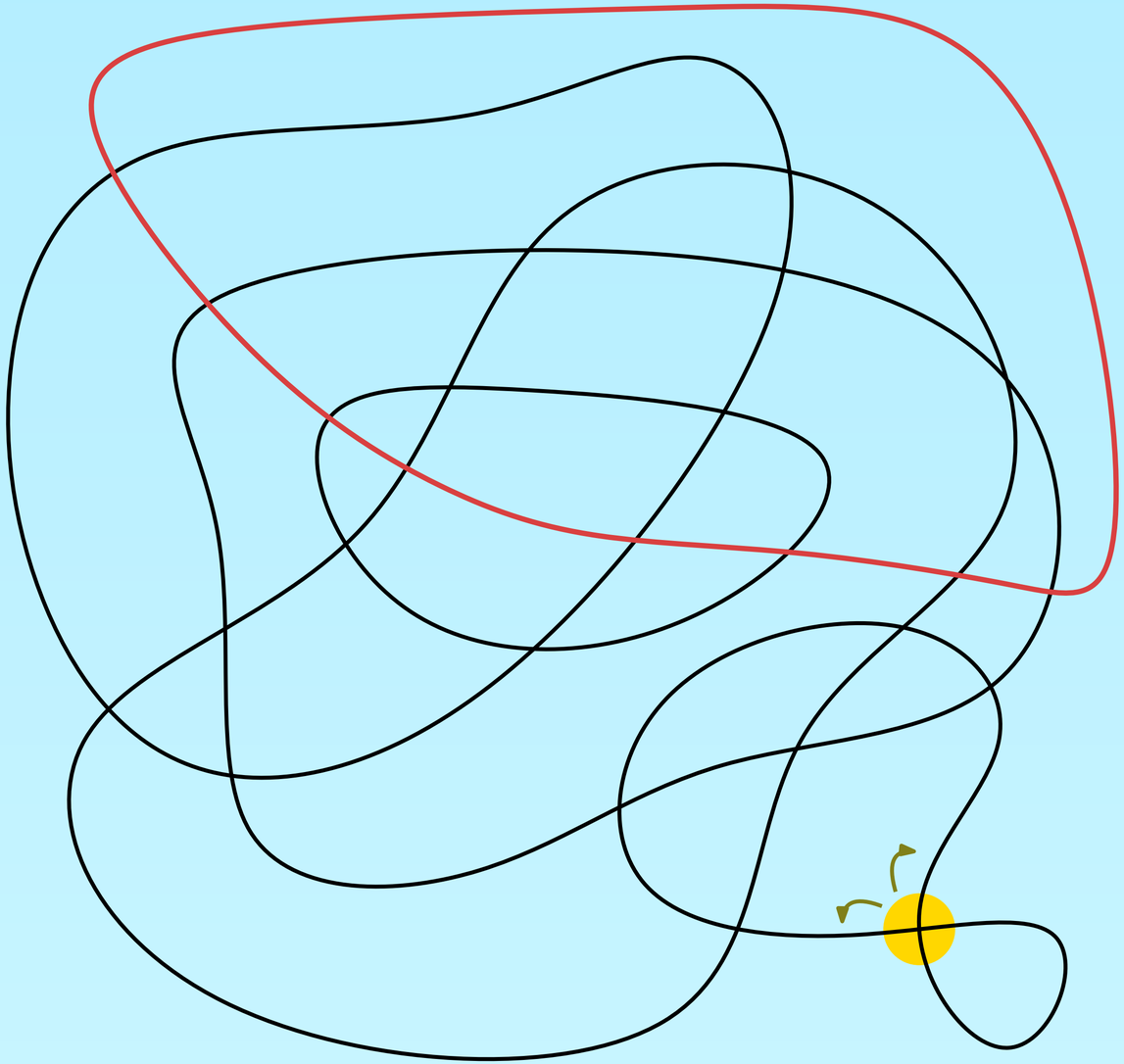
De Nooijer et al. study how popular faces can be removed by adding more curves to the arrangement.



De Nooijer et al. study how popular faces can be removed by adding more curves to the arrangement.

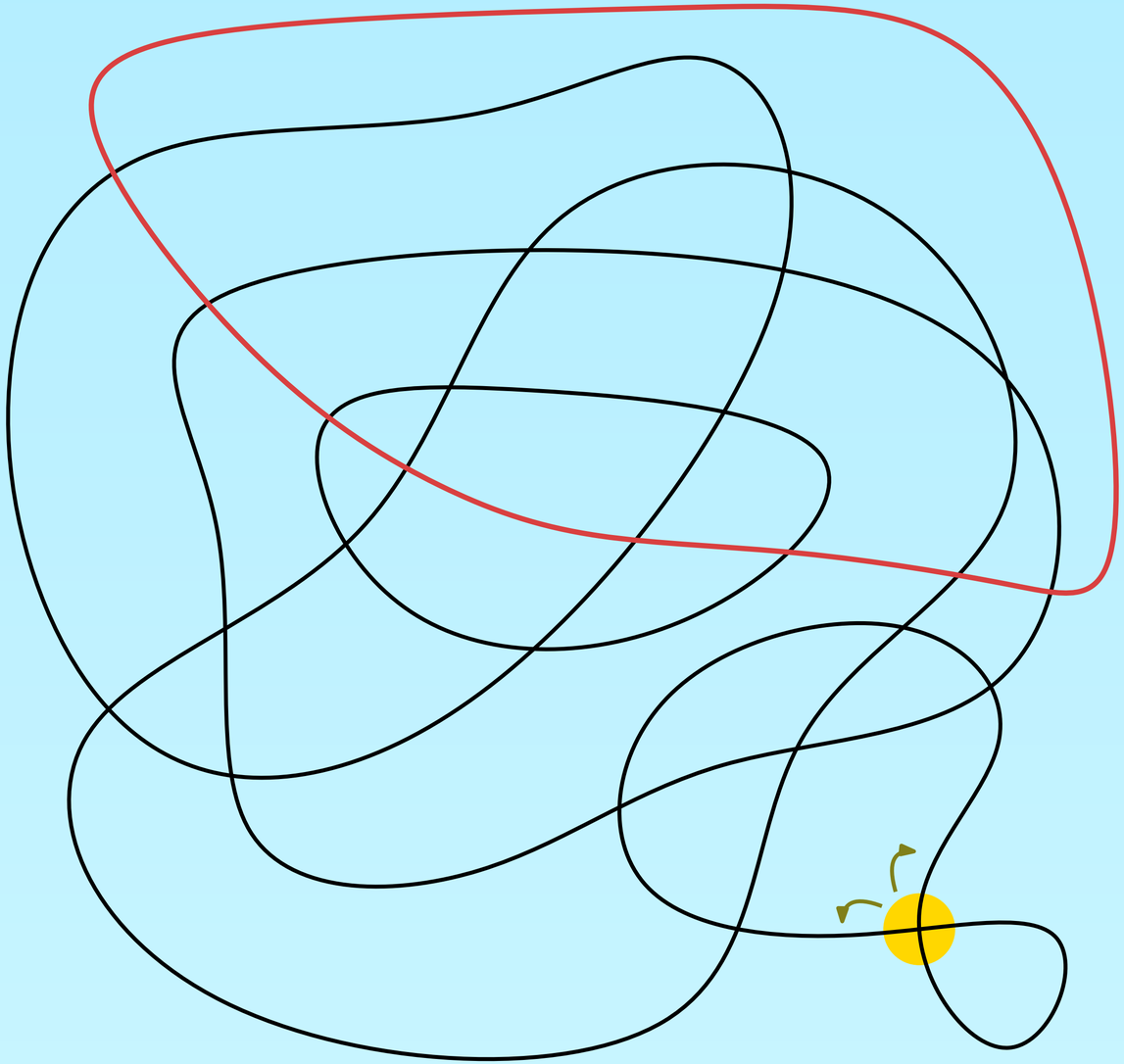


De Nooijer et al. study how popular faces can be removed by adding more curves to the arrangement.



De Nooijer et al. study how popular faces can be removed by adding more curves to the arrangement.

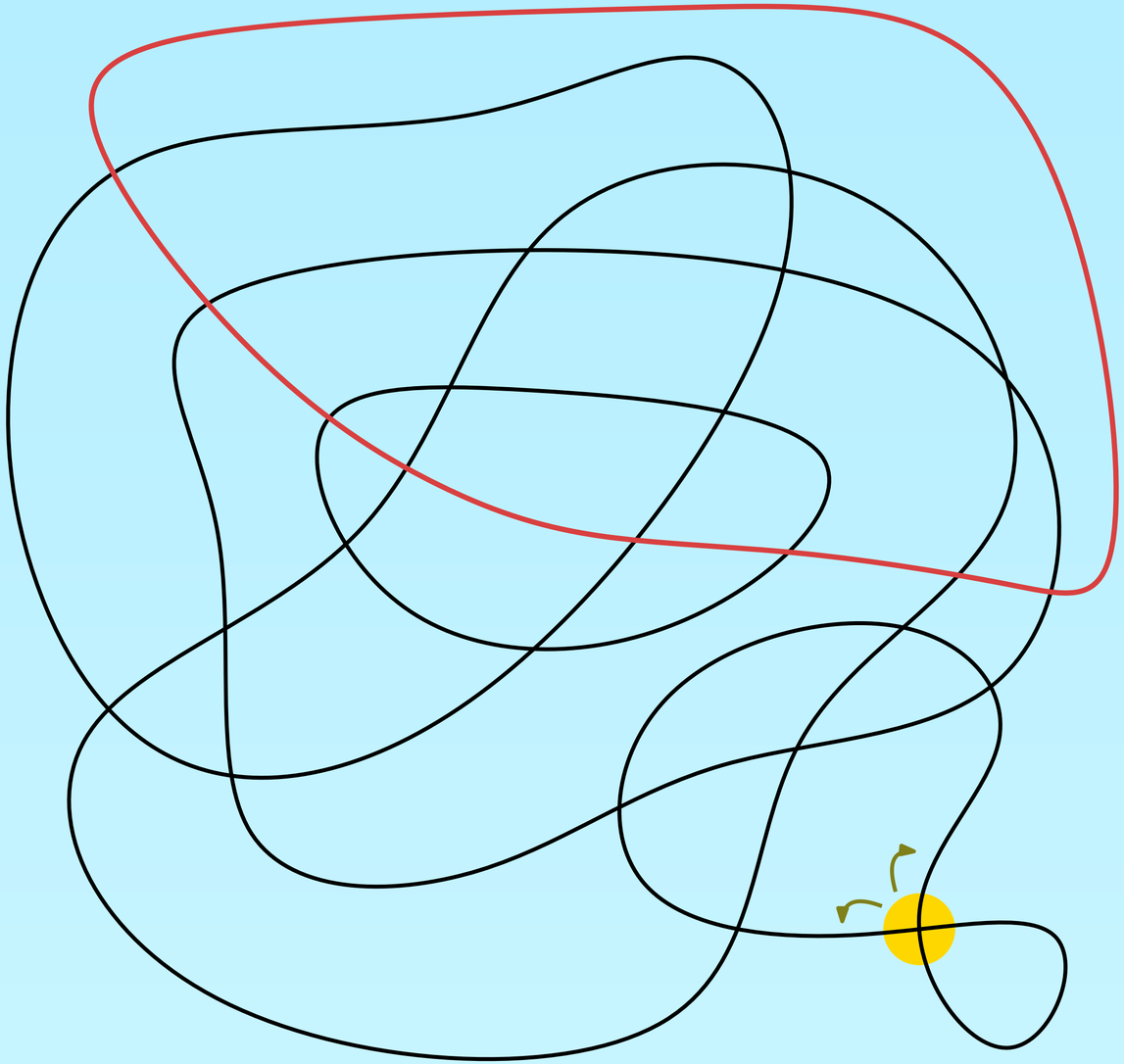
Advantage: does not change geometry of existing curves.



De Nooijer et al. study how popular faces can be removed by adding more curves to the arrangement.

Advantage: does not change geometry of existing curves.

Drawback: arrangement gets more complicated, and faces get smaller.



De Nooijer et al. study how popular faces can be removed by adding more curves to the arrangement.

Advantage: does not change geometry of existing curves.

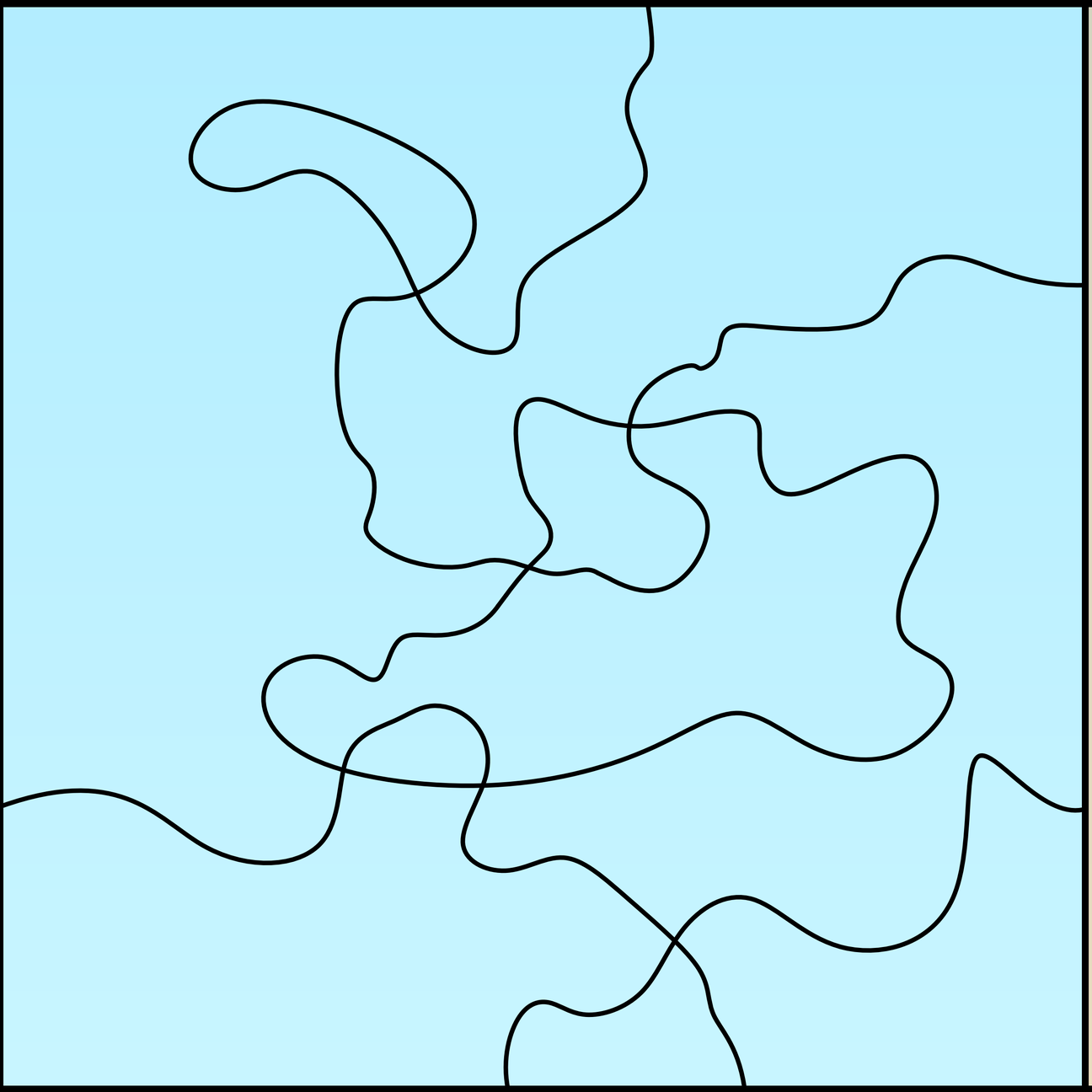
Drawback: arrangement gets more complicated, and faces get smaller.

Drawback: self-intersections cannot be removed this way.

Idea: change combinatorics, not geometry.

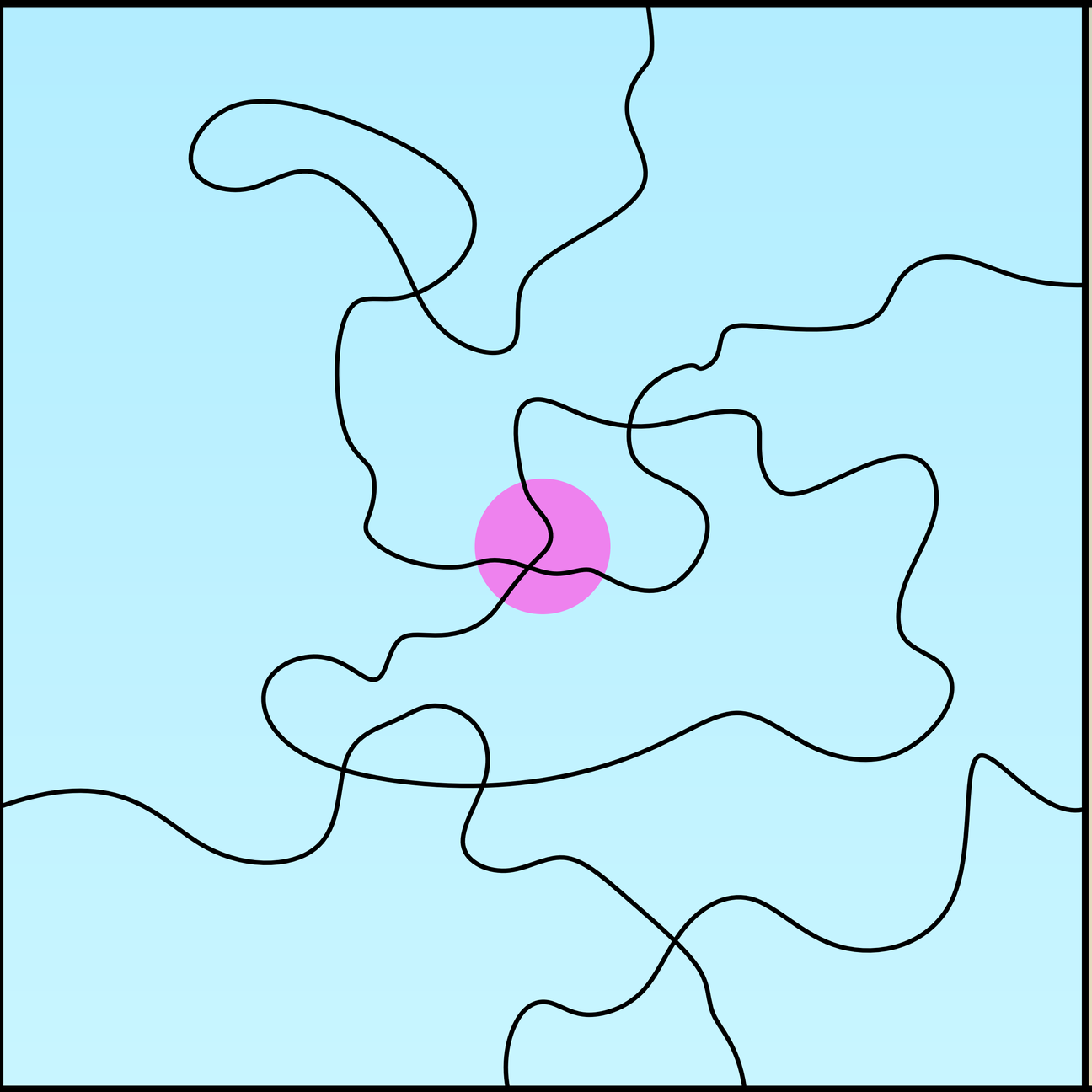
Idea: change combinatorics, not geometry.

A switch is a disk of radius ε whose boundary intersects the curve arrangement exactly four times.



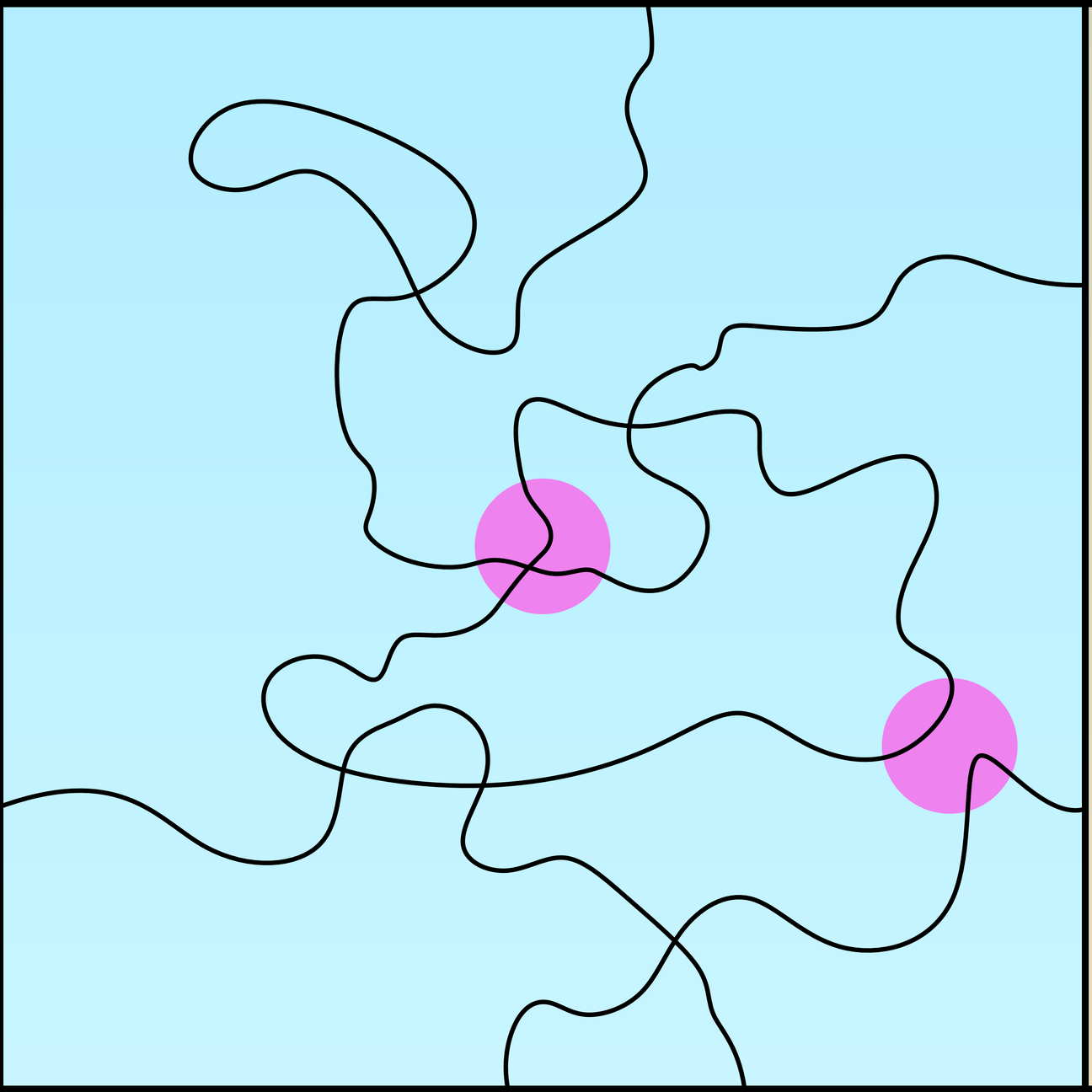
Idea: change combinatorics, not geometry.

A switch is a disk of radius ε whose boundary intersects the curve arrangement exactly four times.



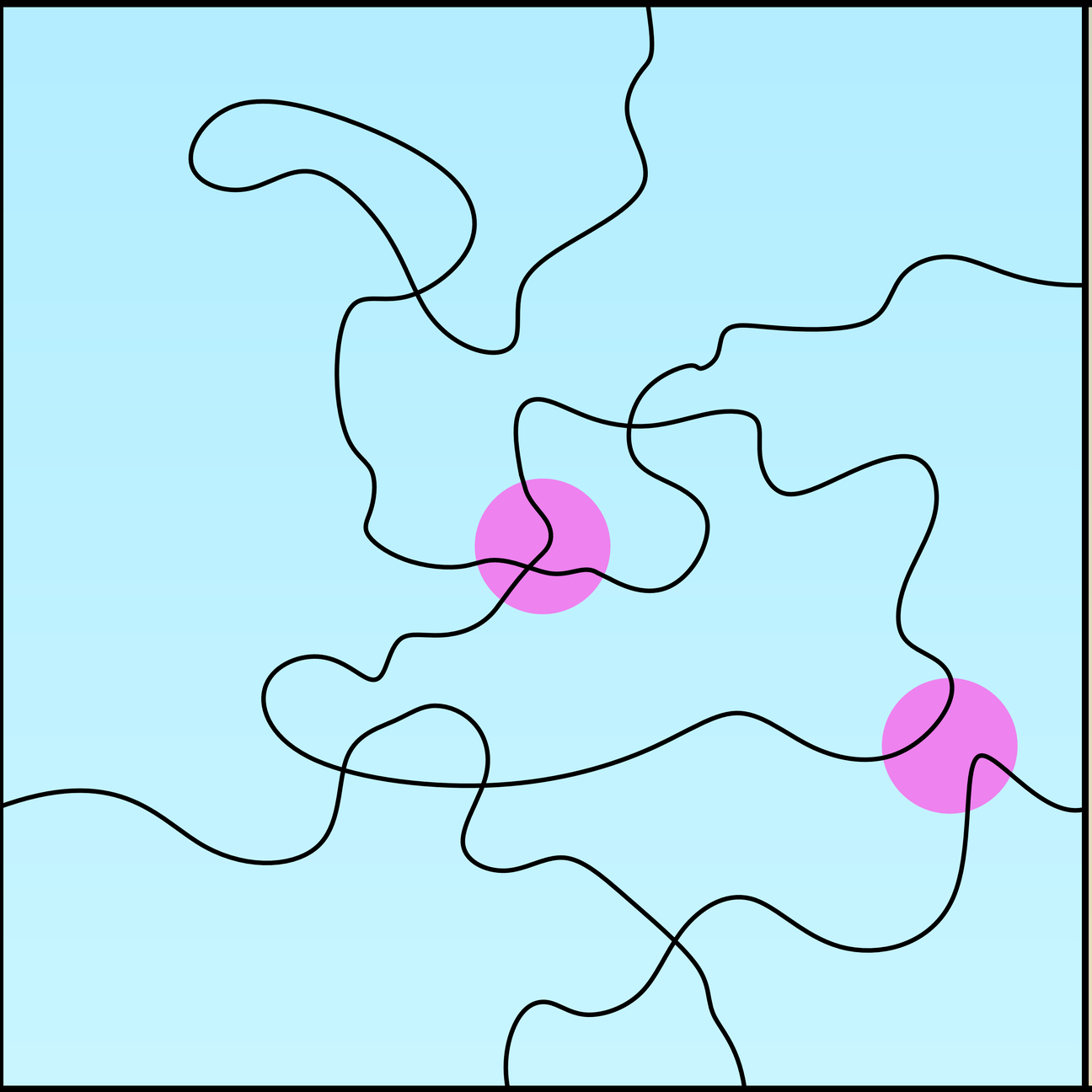
Idea: change combinatorics, not geometry.

A switch is a disk of radius ε whose boundary intersects the curve arrangement exactly four times.



Idea: change combinatorics, not geometry.

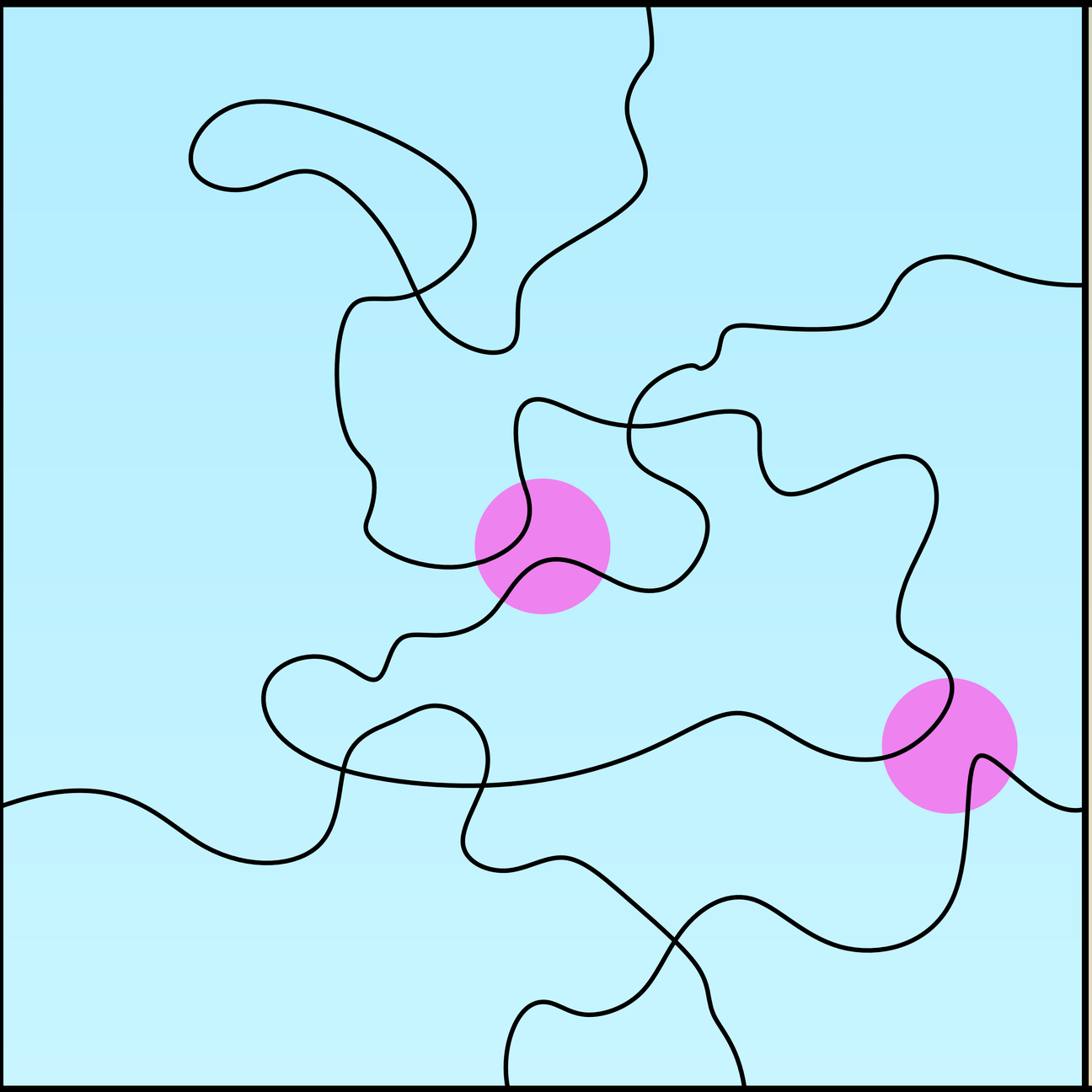
A switch is a disk of radius ε whose boundary intersects the curve arrangement exactly four times.



Idea: change combinatorics, not geometry.

A switch is a disk of radius ε whose boundary intersects the curve arrangement exactly four times.

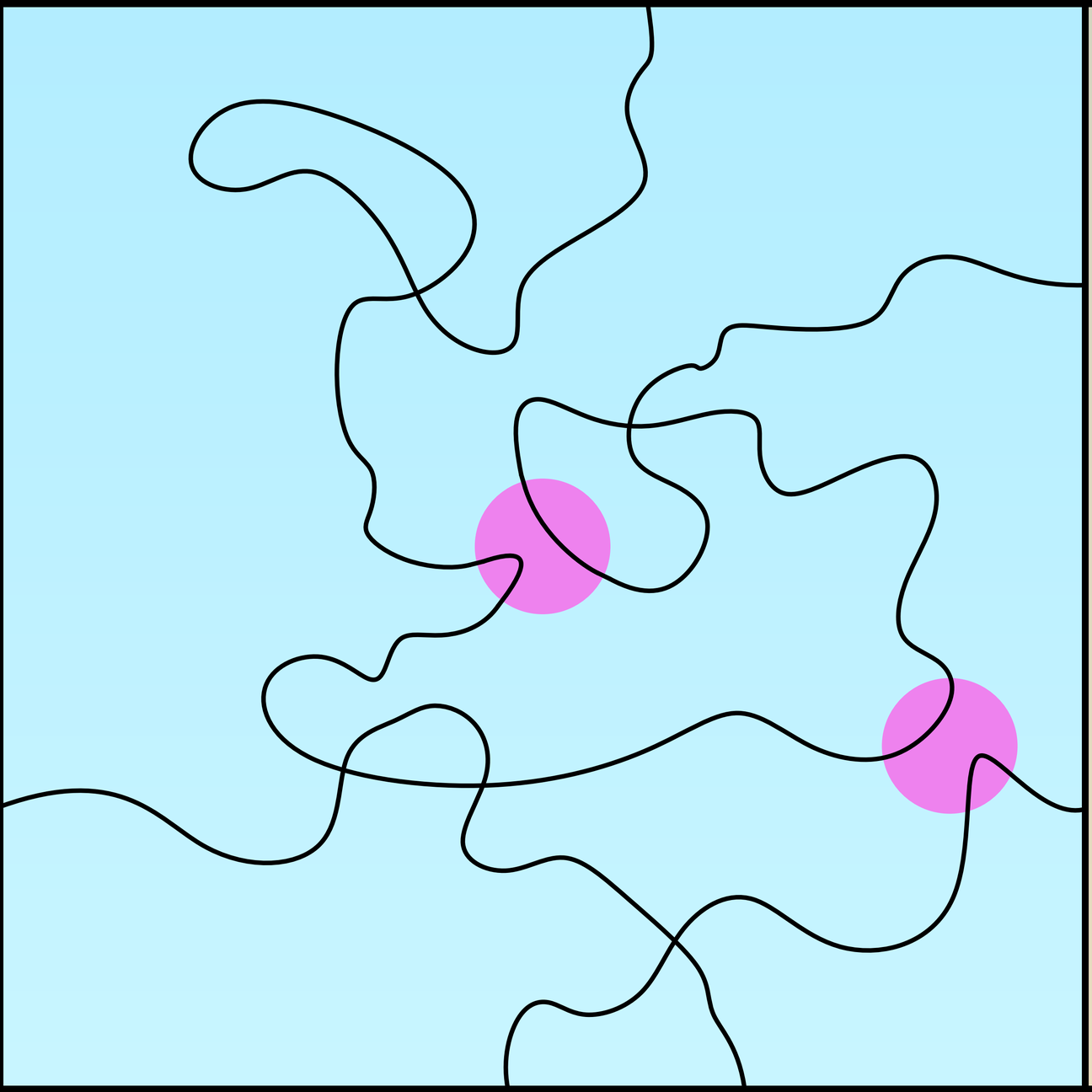
We can reroute the curves inside a switch in three ways.



Idea: change combinatorics, not geometry.

A switch is a disk of radius ε whose boundary intersects the curve arrangement exactly four times.

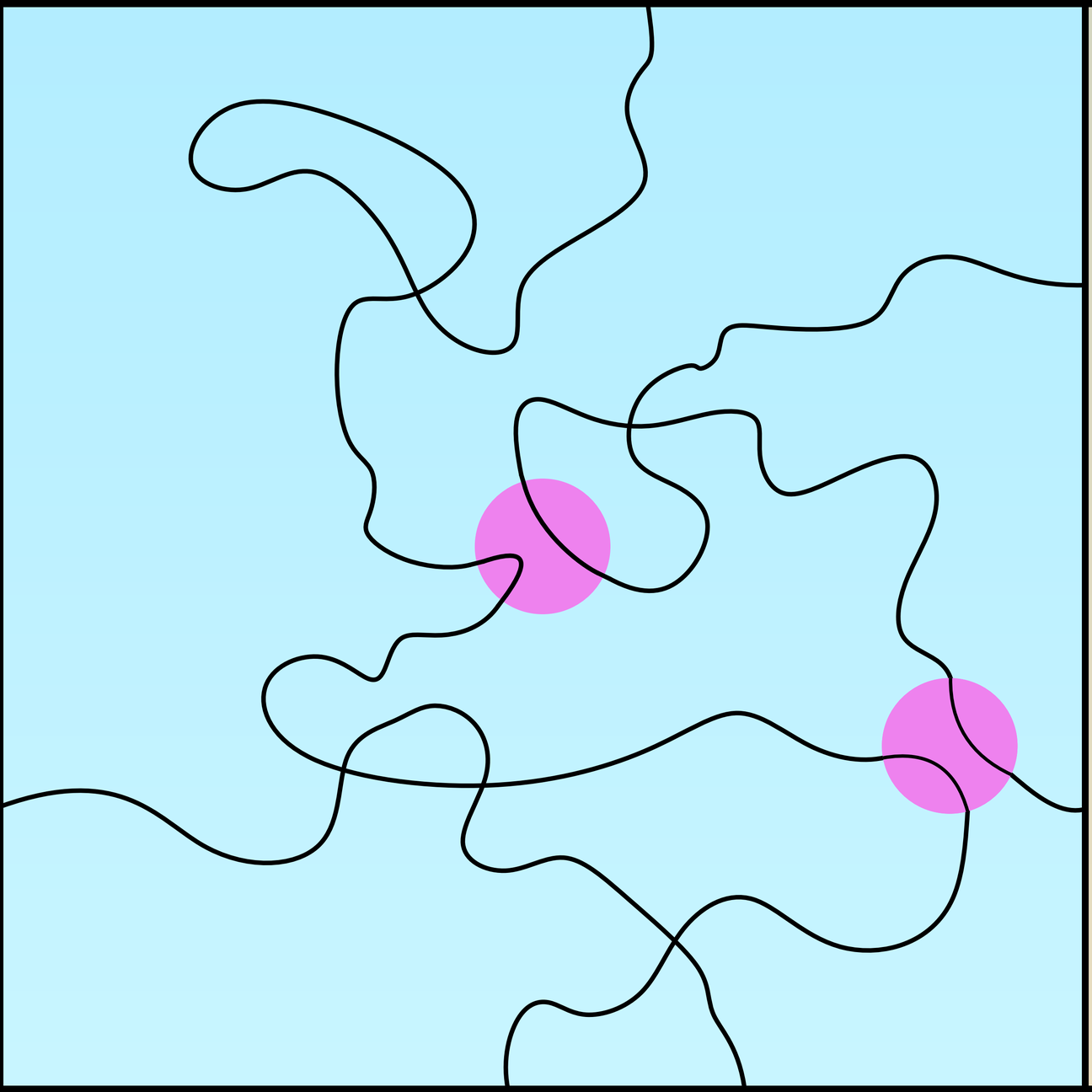
We can reroute the curves inside a switch in three ways.



Idea: change combinatorics, not geometry.

A switch is a disk of radius ε whose boundary intersects the curve arrangement exactly four times.

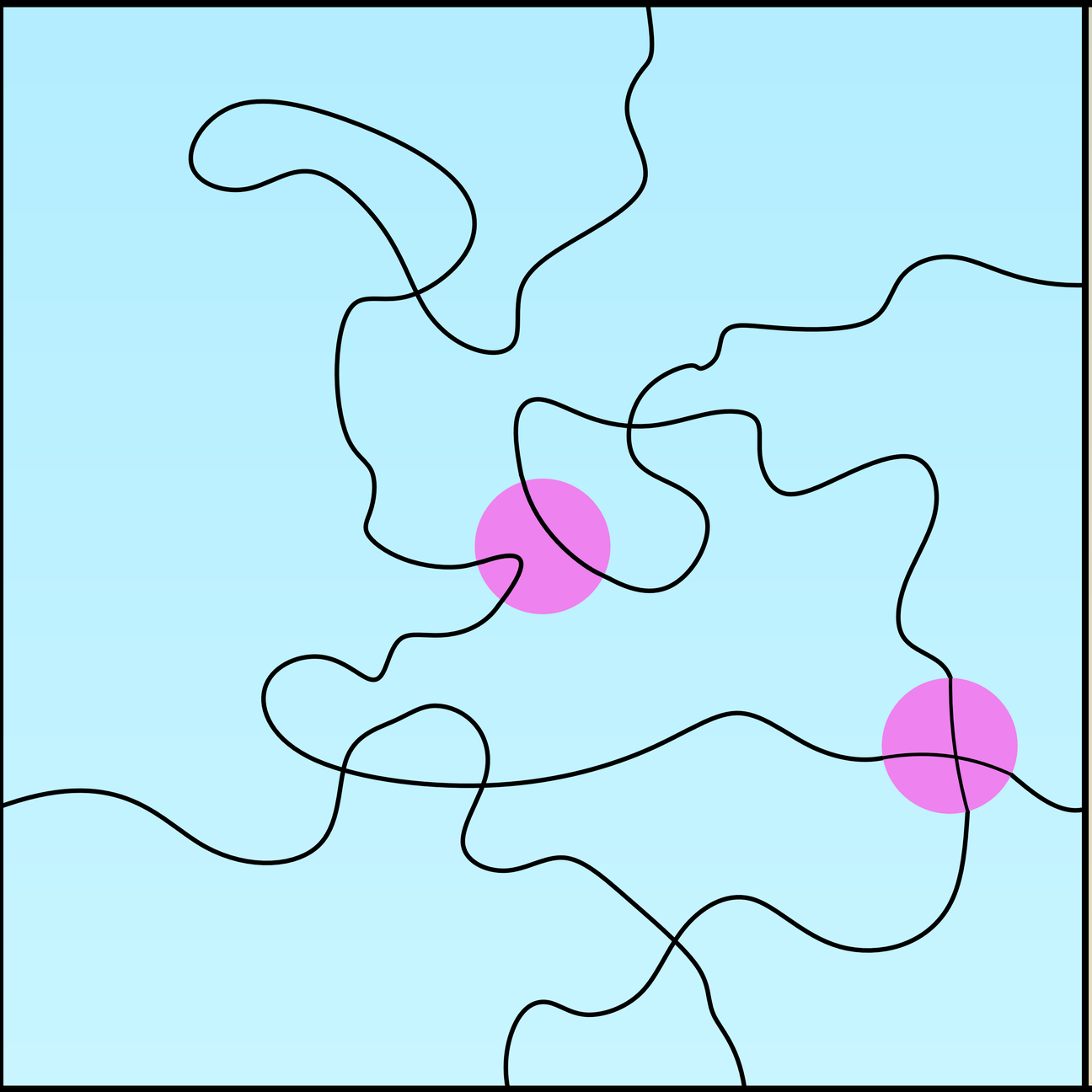
We can reroute the curves inside a switch in three ways.



Idea: change combinatorics, not geometry.

A switch is a disk of radius ε whose boundary intersects the curve arrangement exactly four times.

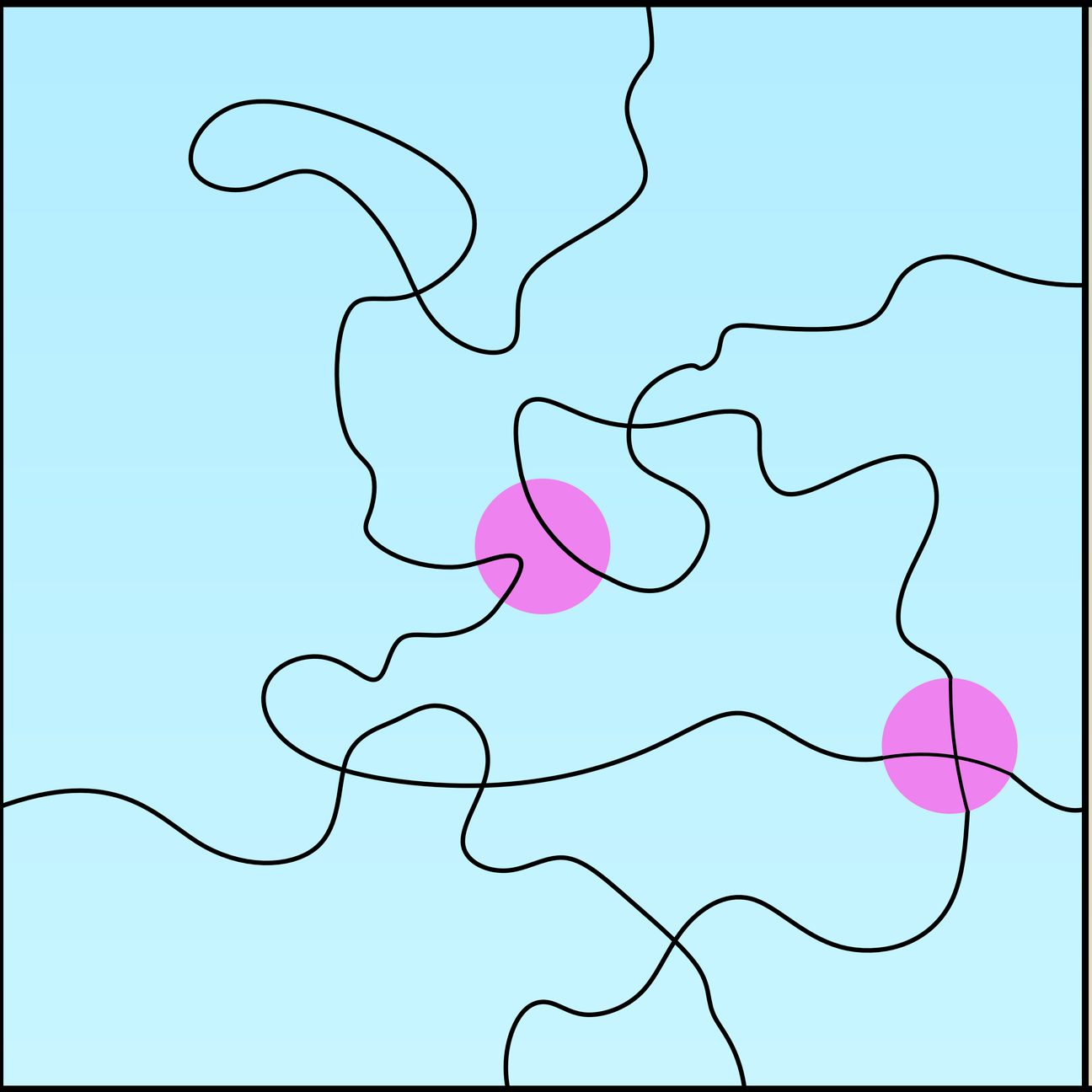
We can reroute the curves inside a switch in three ways.



Idea: change combinatorics, not geometry.

A switch is a disk of radius ε whose boundary intersects the curve arrangement exactly four times.

We can reroute the curves inside a switch in three ways.

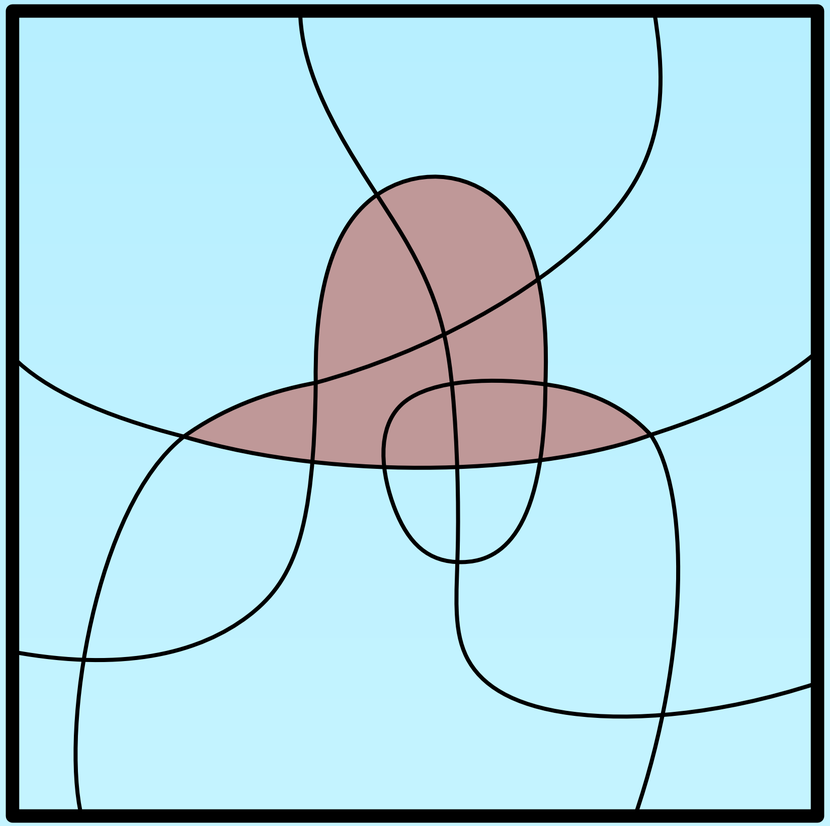


Idea: change combinatorics, not geometry.

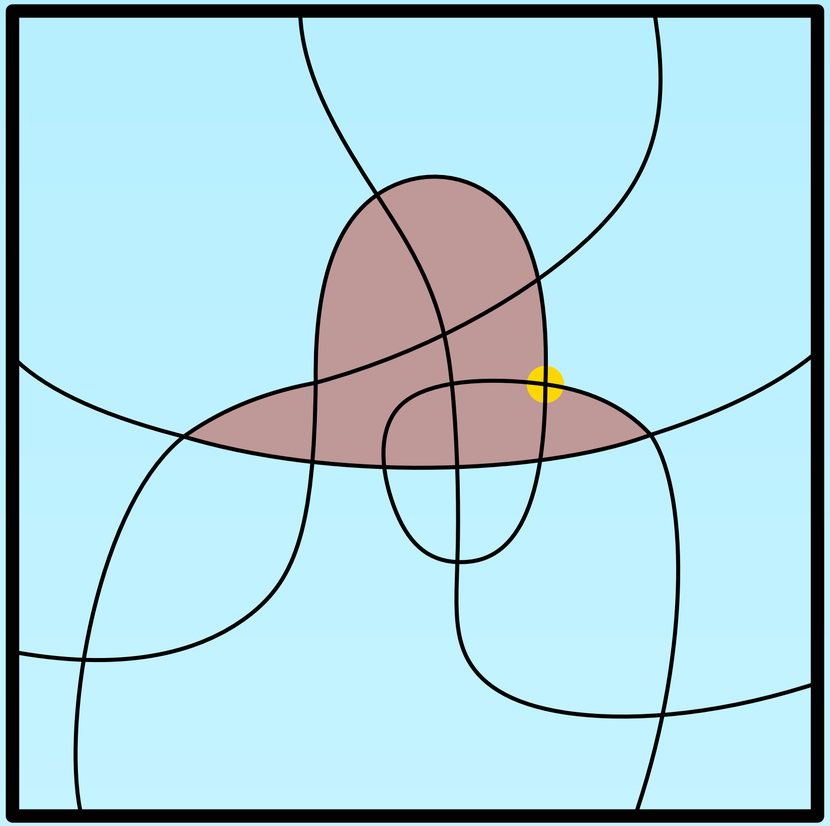
A switch is a disk of radius ε whose boundary intersects the curve arrangement exactly four times.

We can reroute the curves inside a switch in three ways.

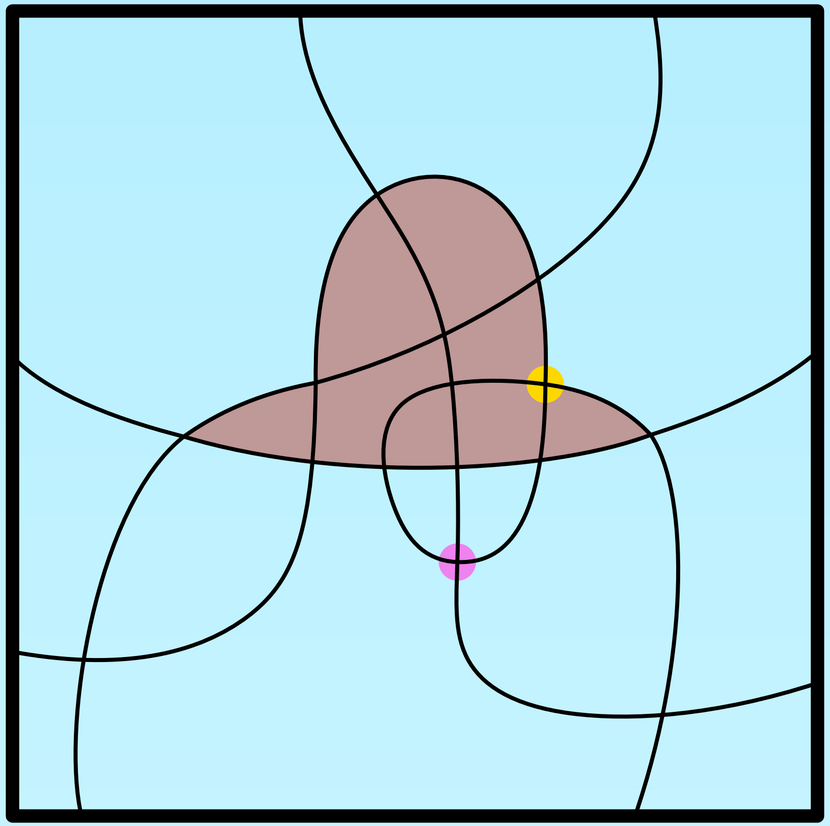
Only limited impact on overall aesthetics, large impact on curve structure.



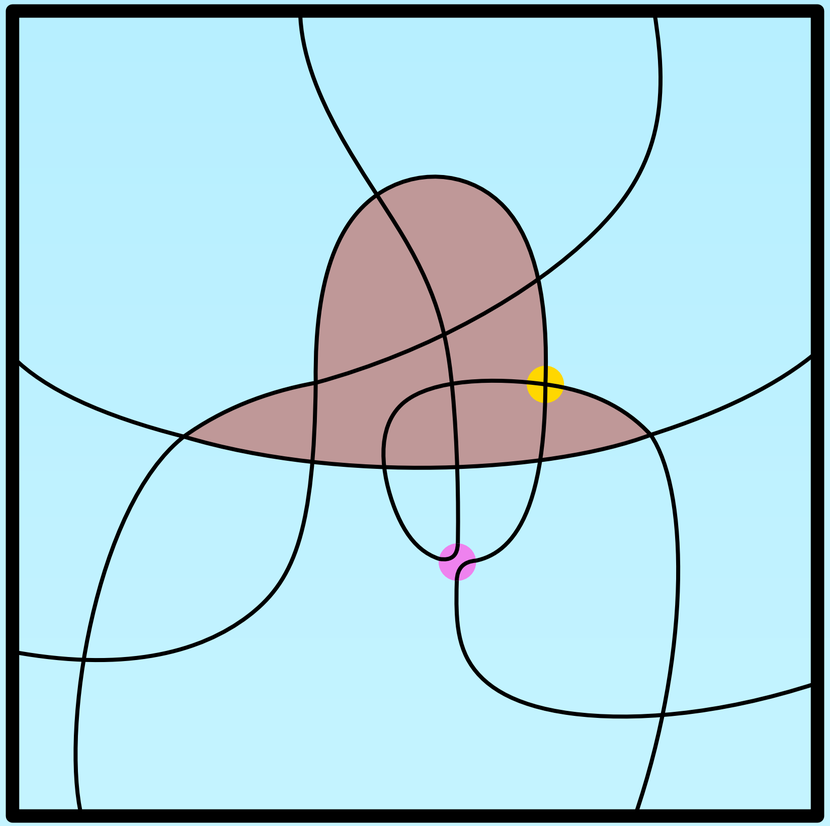
Allows turning expert nonograms into advanced ones, or advanced into basic.



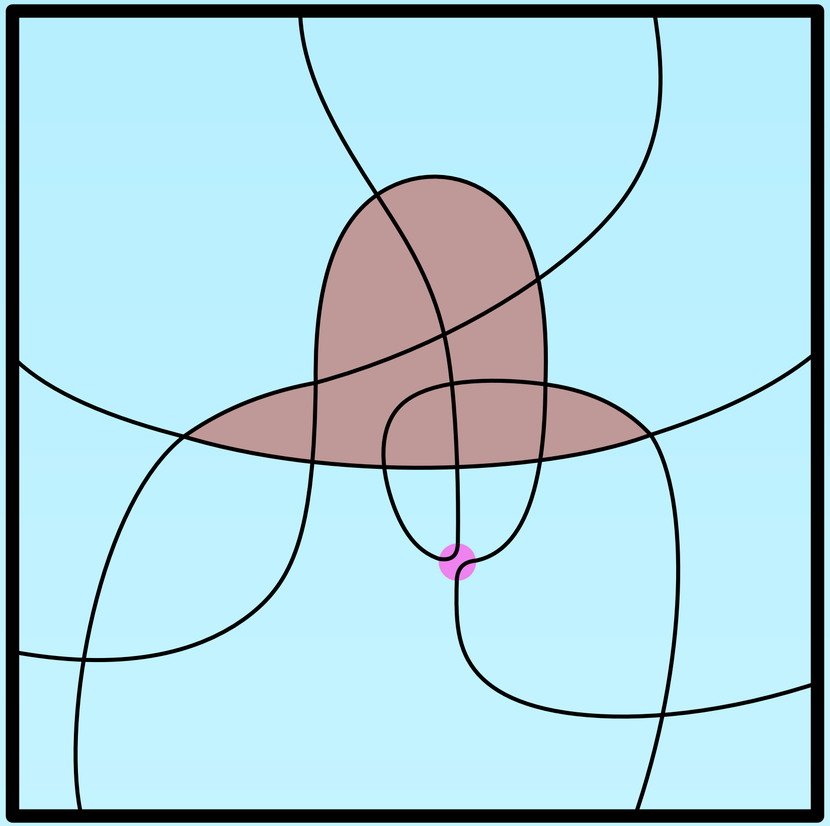
Allows turning expert nonograms into advanced ones, or advanced into basic.



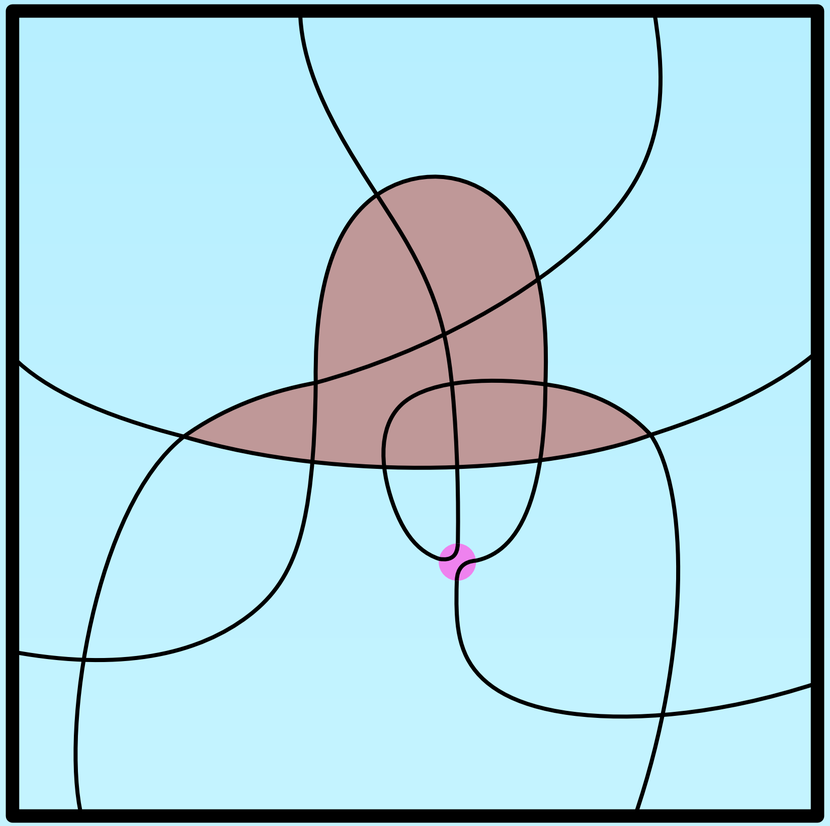
Allows turning expert nonograms into advanced ones, or advanced into basic.



Allows turning expert nonograms into advanced ones, or advanced into basic.

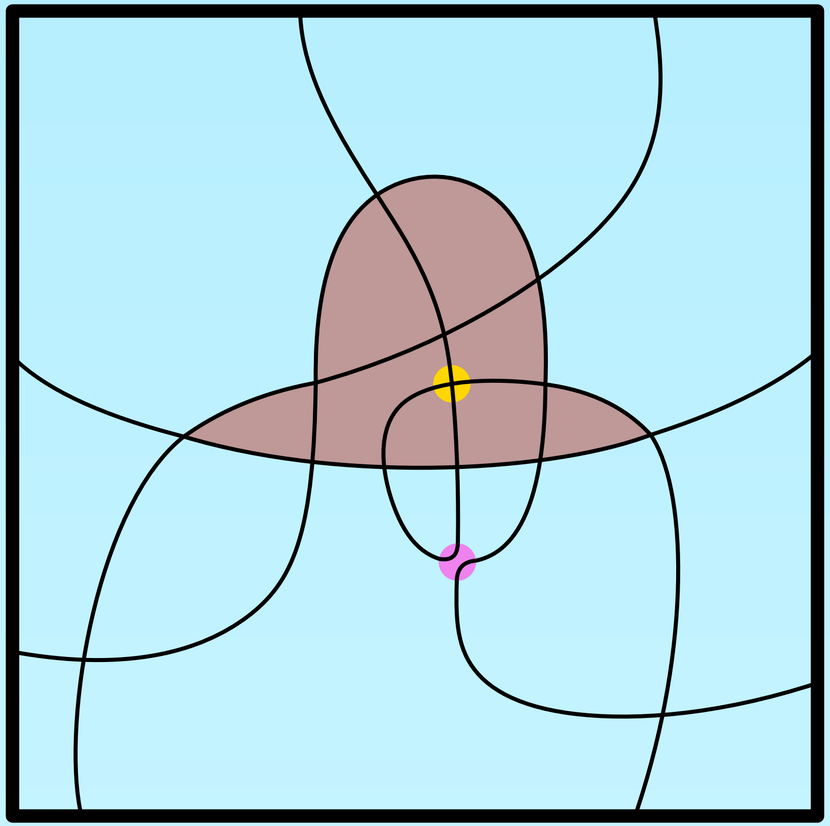


Allows turning expert nonograms into advanced ones, or advanced into basic.



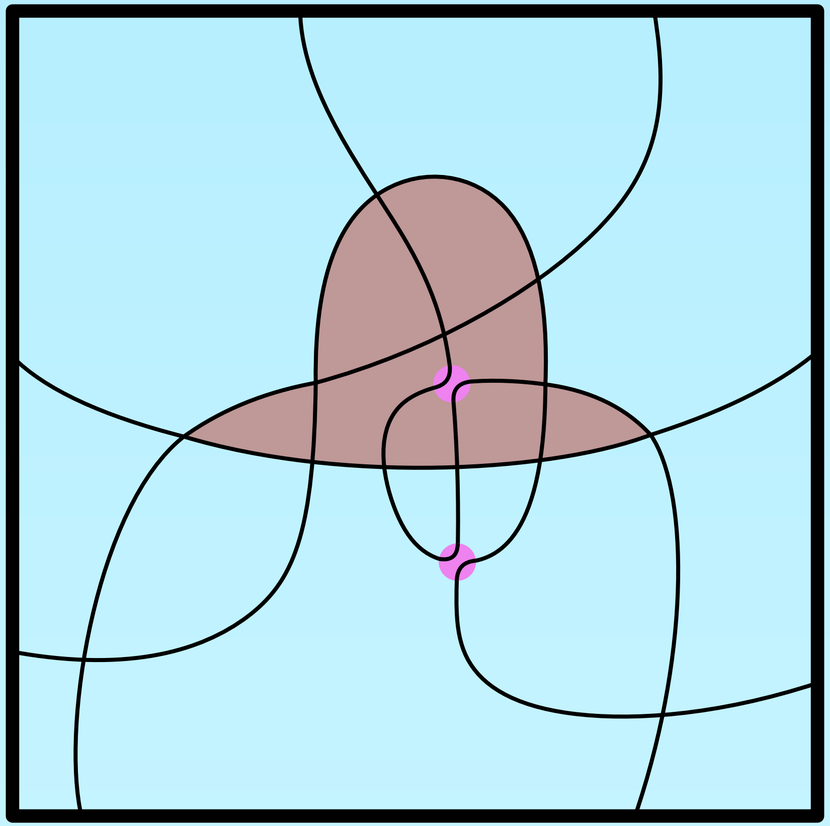
Allows turning expert nonograms into advanced ones, or advanced into basic.

Note that we can remove self-intersections either by switching other vertices, or the vertex itself.



Allows turning expert nonograms into advanced ones, or advanced into basic.

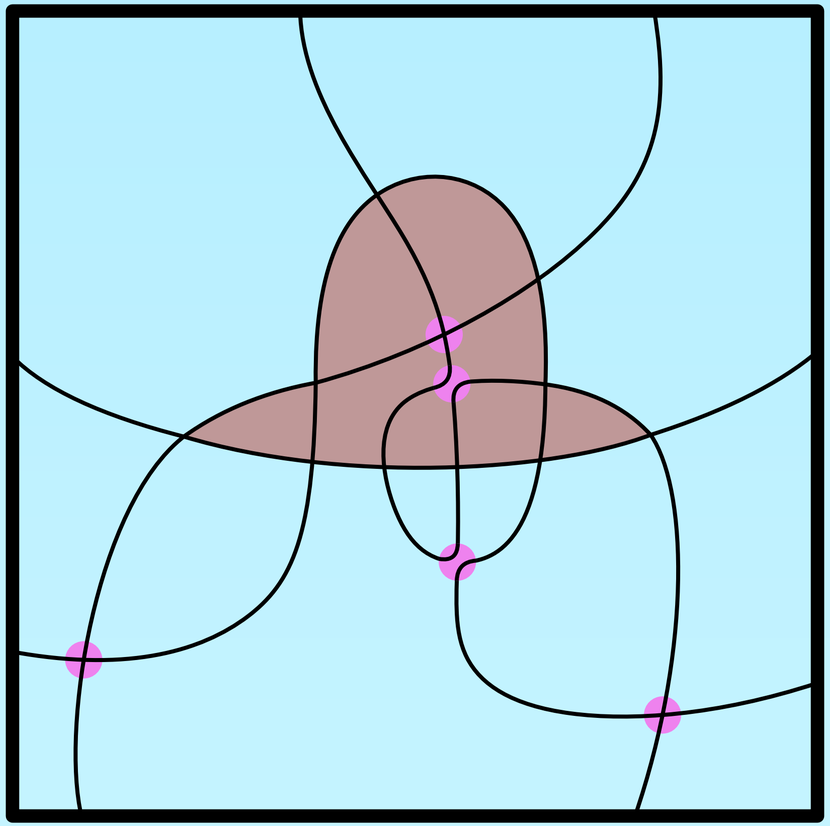
Note that we can remove self-intersections either by switching other vertices, or the vertex itself.



Allows turning expert nonograms into advanced ones, or advanced into basic.

Note that we can remove self-intersections either by switching other vertices, or the vertex itself.

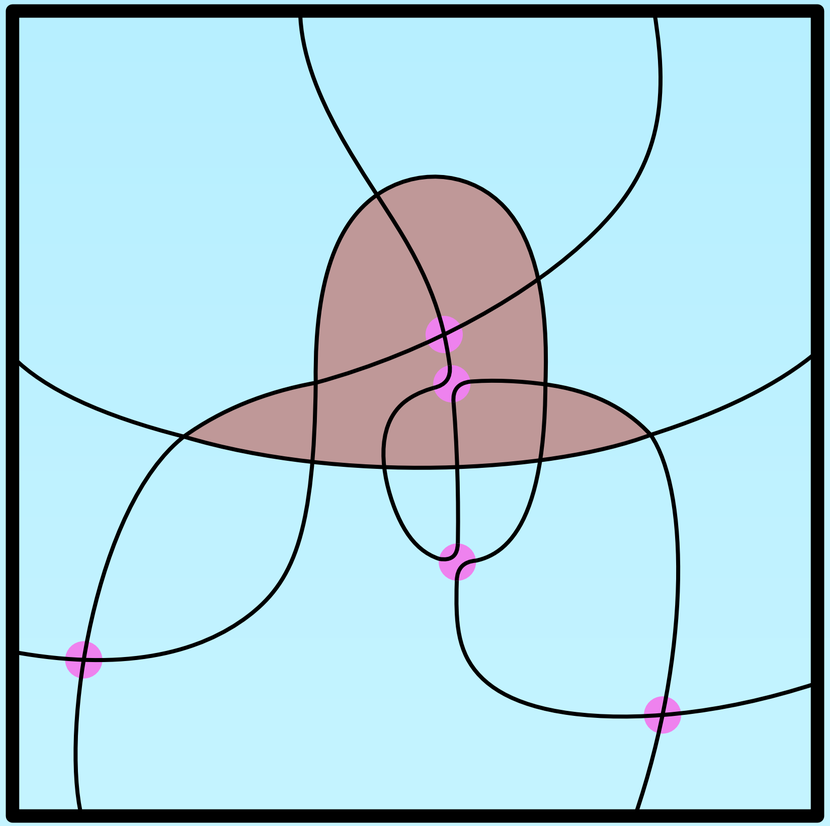
Not every switch is valid in the context of nonograms!



Allows turning expert nonograms into advanced ones, or advanced into basic.

Note that we can remove self-intersections either by switching other vertices, or the vertex itself.

Not every switch is valid in the context of nonograms!

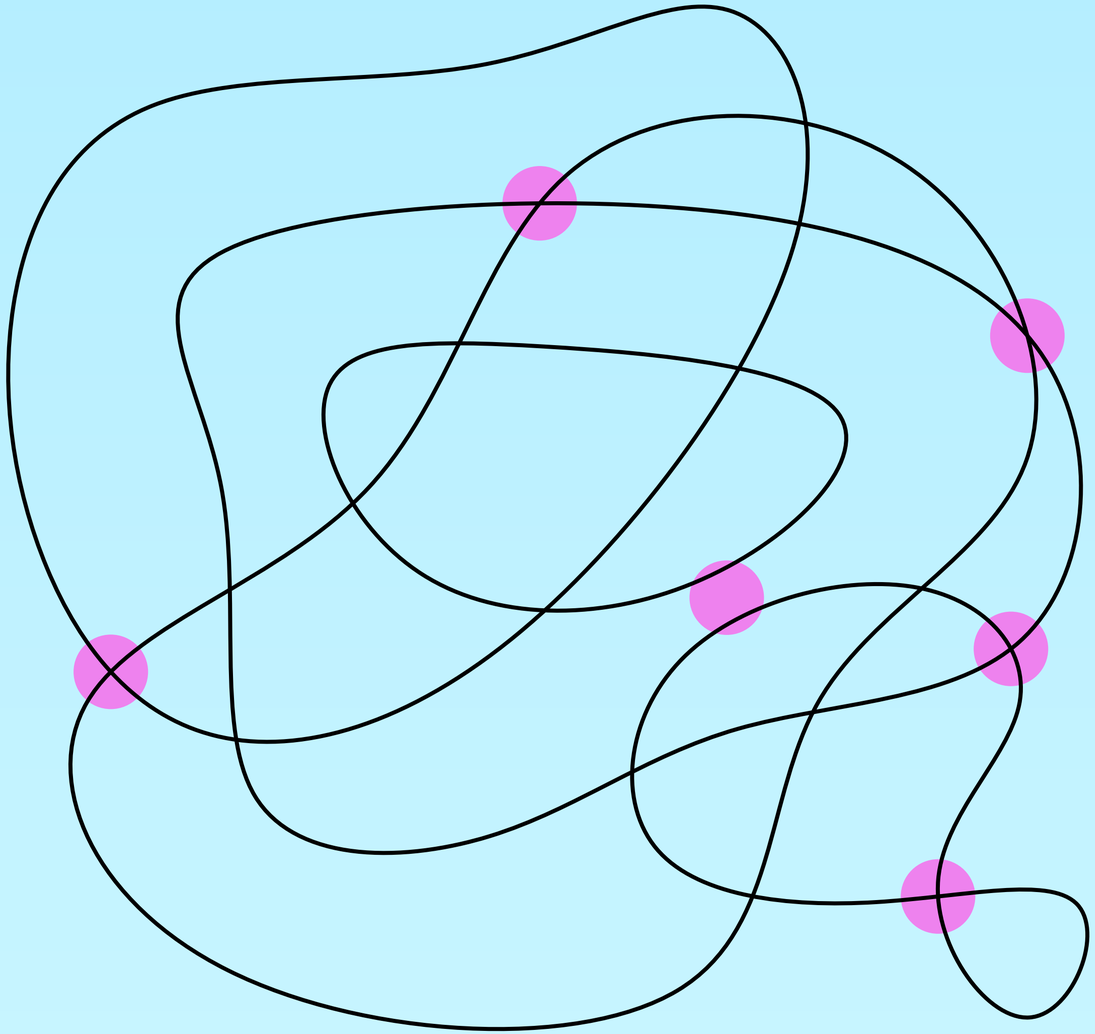


Allows turning expert nonograms into advanced ones, or advanced into basic.

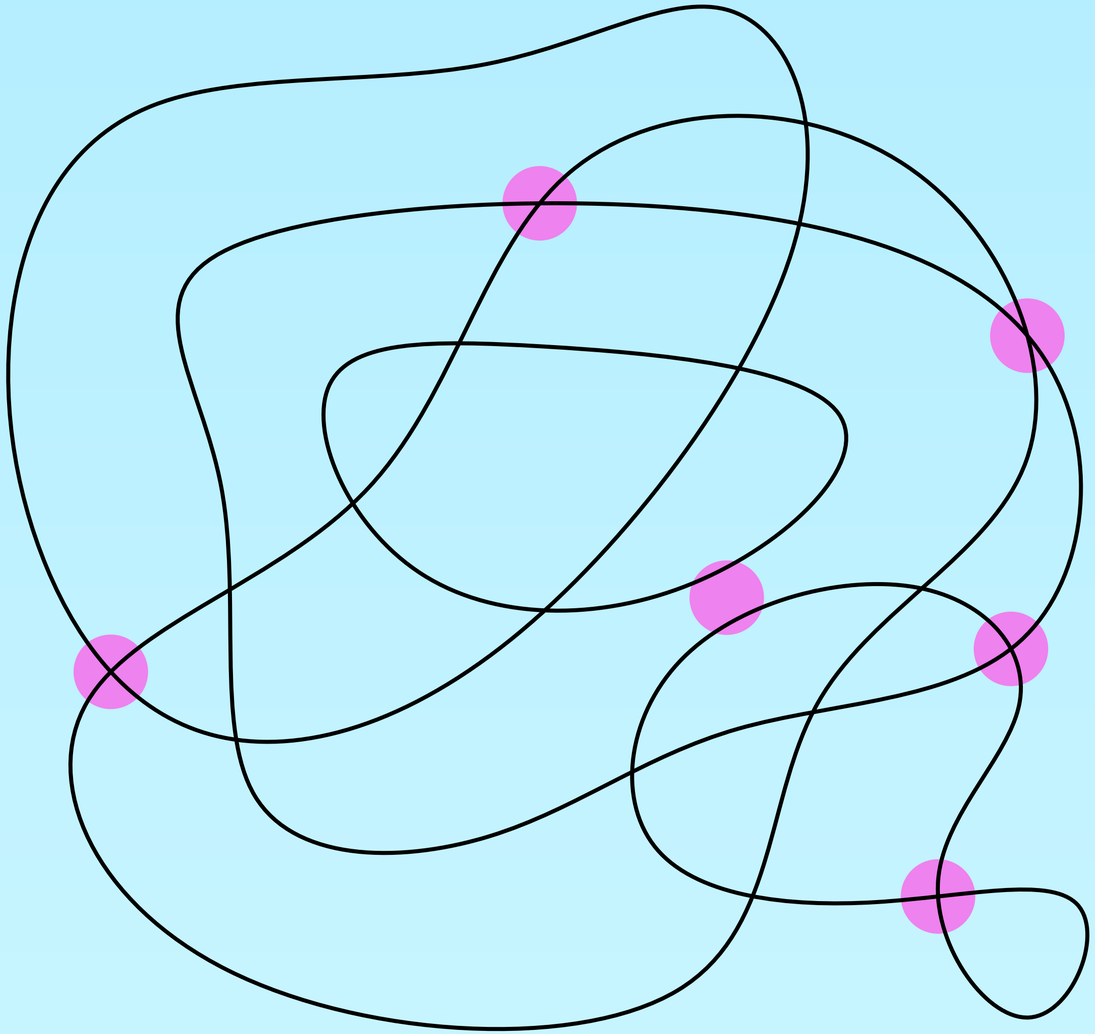
Note that we can remove self-intersections either by switching other vertices, or the vertex itself.

Not every switch is valid in the context of nonograms!

Also, the solvability of the puzzle may change.

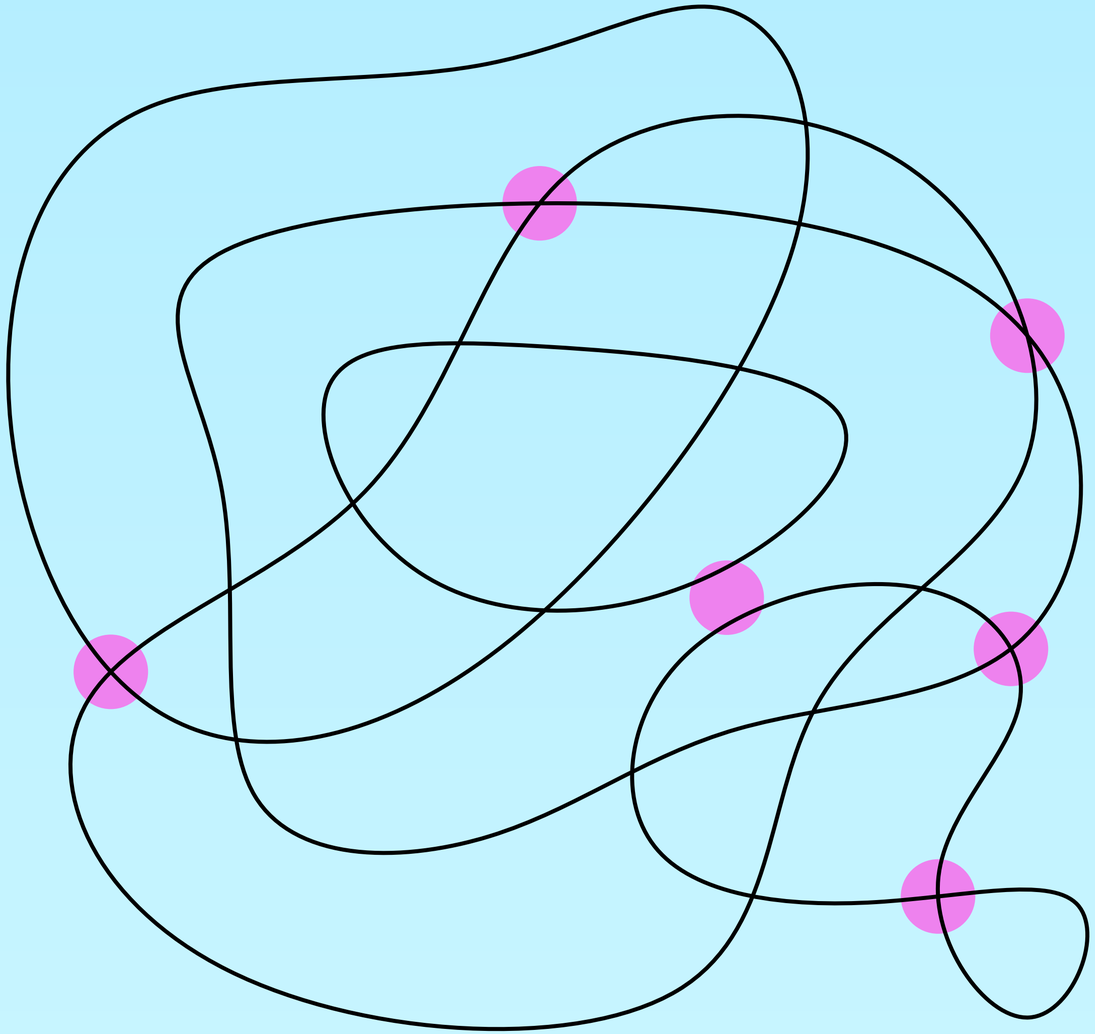


A switchable arrangement is an arrangement of closed curves in the plane, together with a set of switches.



A switchable arrangement is an arrangement of closed curves in the plane, together with a set of switches.

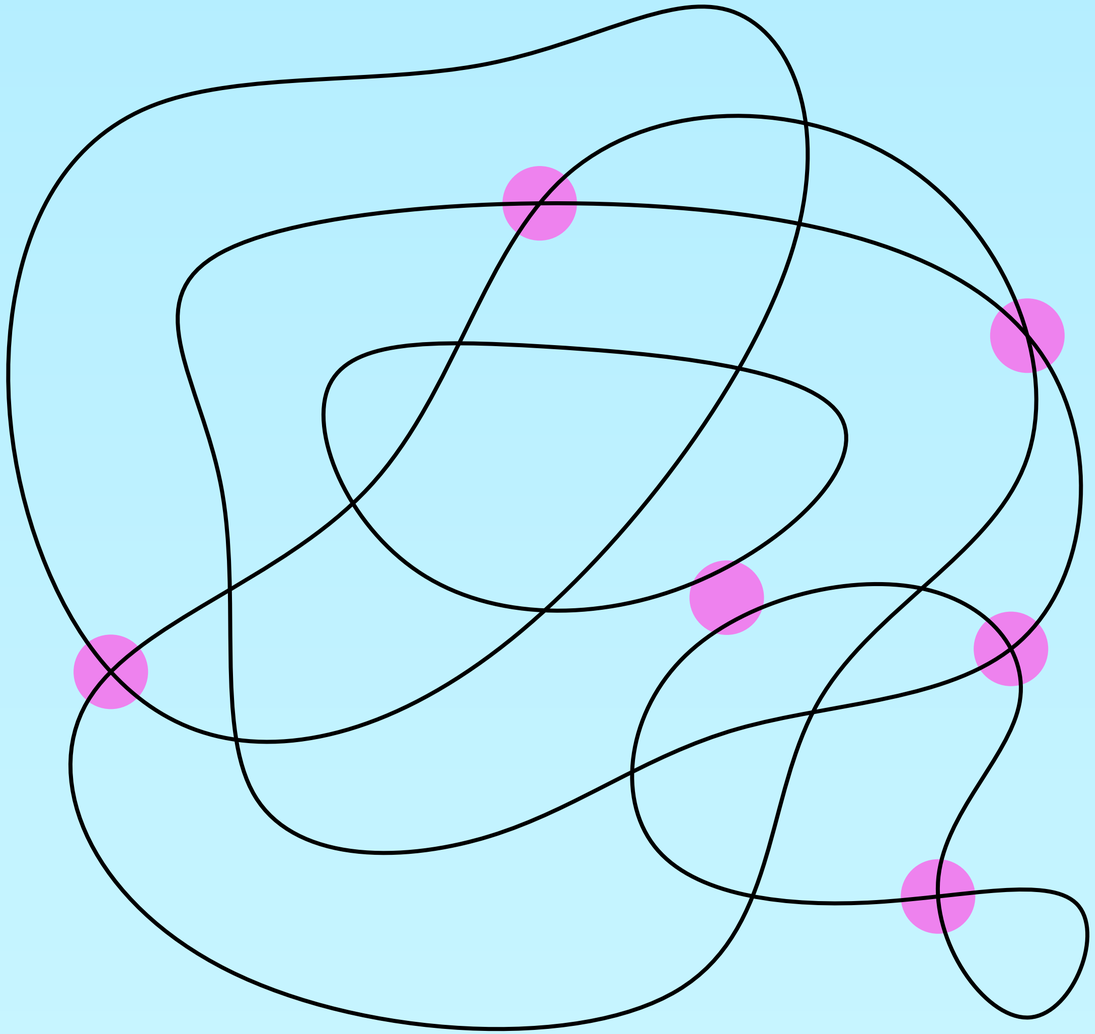
(We don't worry about how to obtain the set of switches...



A switchable arrangement is an arrangement of closed curves in the plane, together with a set of switches.

(We don't worry about how to obtain the set of switches...

...and we ignore the frame.)

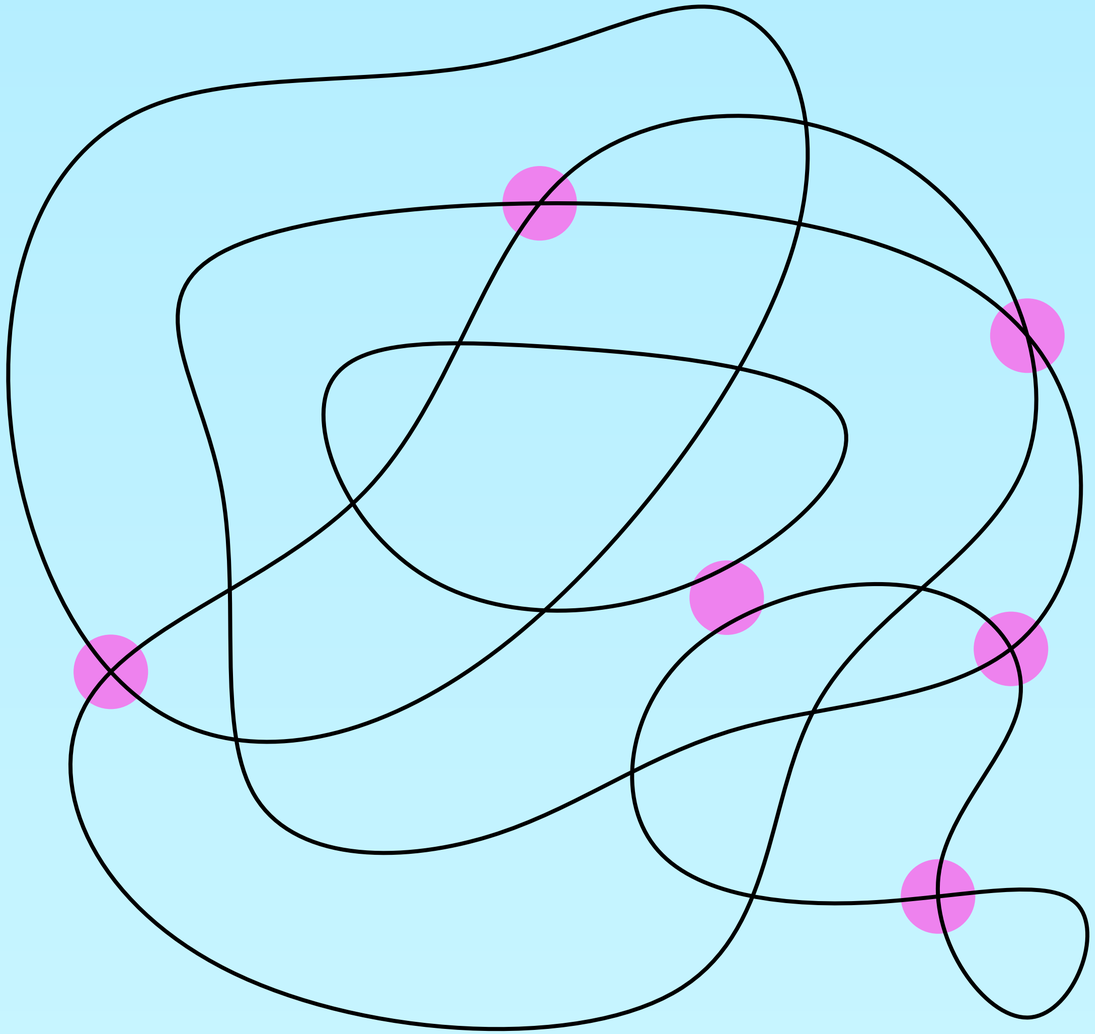


A switchable arrangement is an arrangement of closed curves in the plane, together with a set of switches.

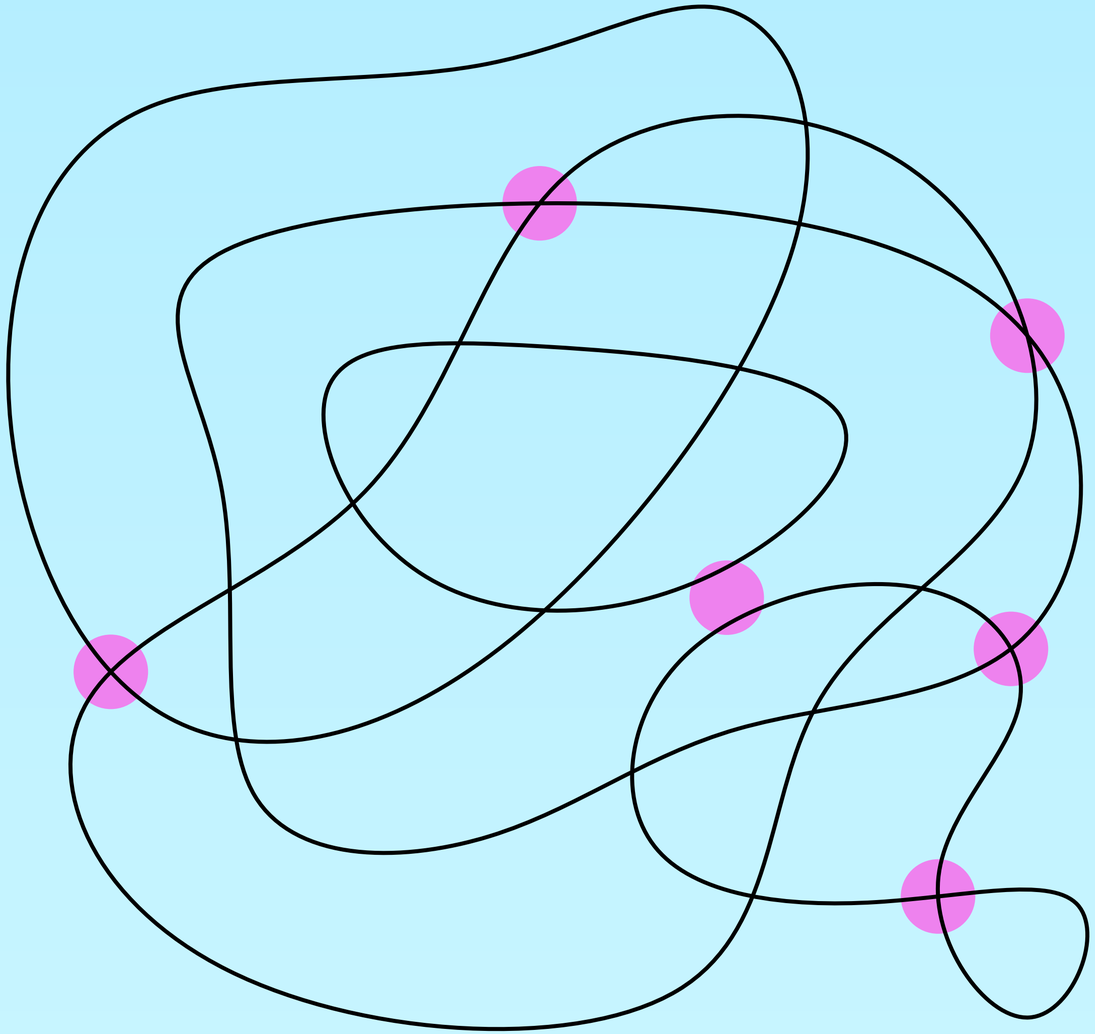
(We don't worry about how to obtain the set of switches...

...and we ignore the frame.)

Now, we have a clean mathematical problem!

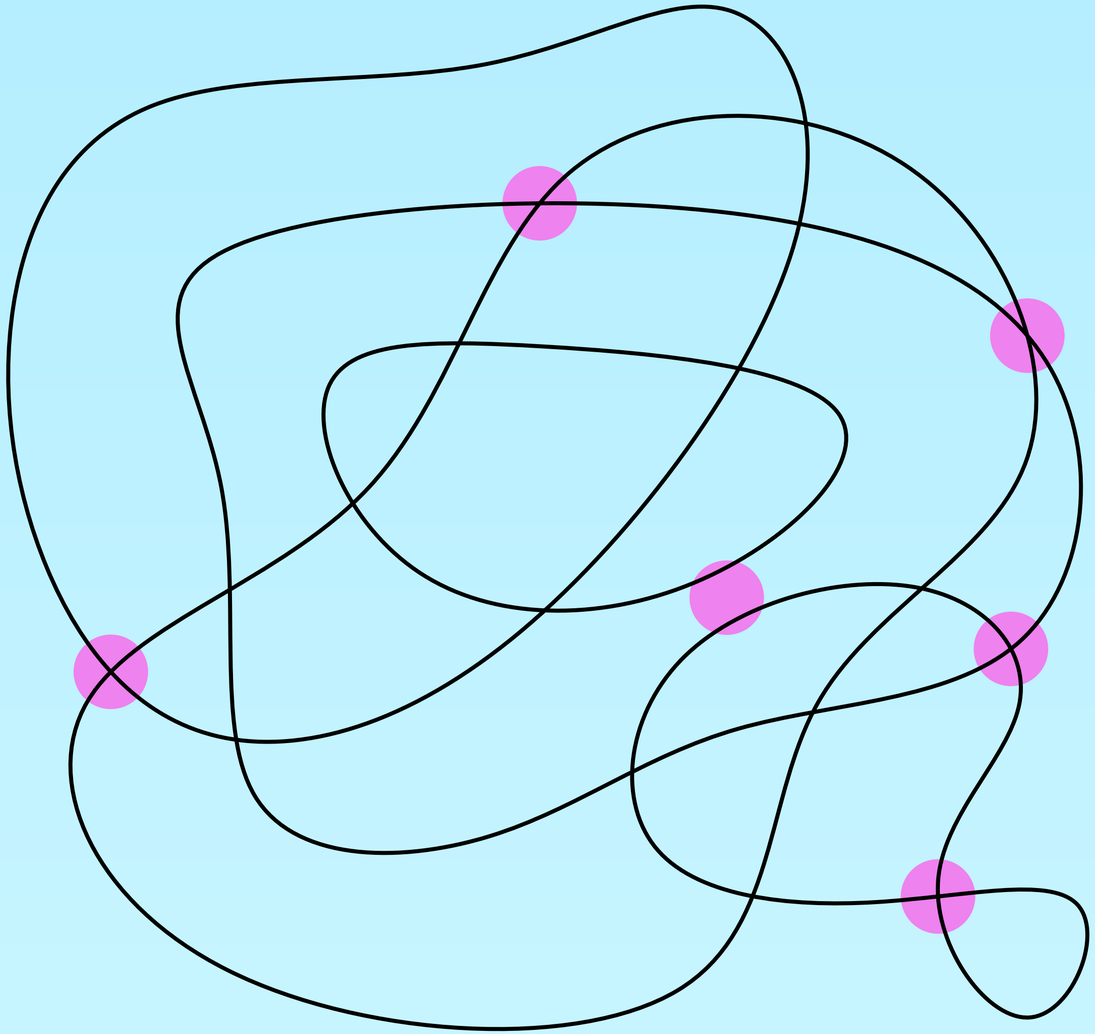


Problem statement.
Given a switchable curve arrangement, can we reconfigure the switches such that:



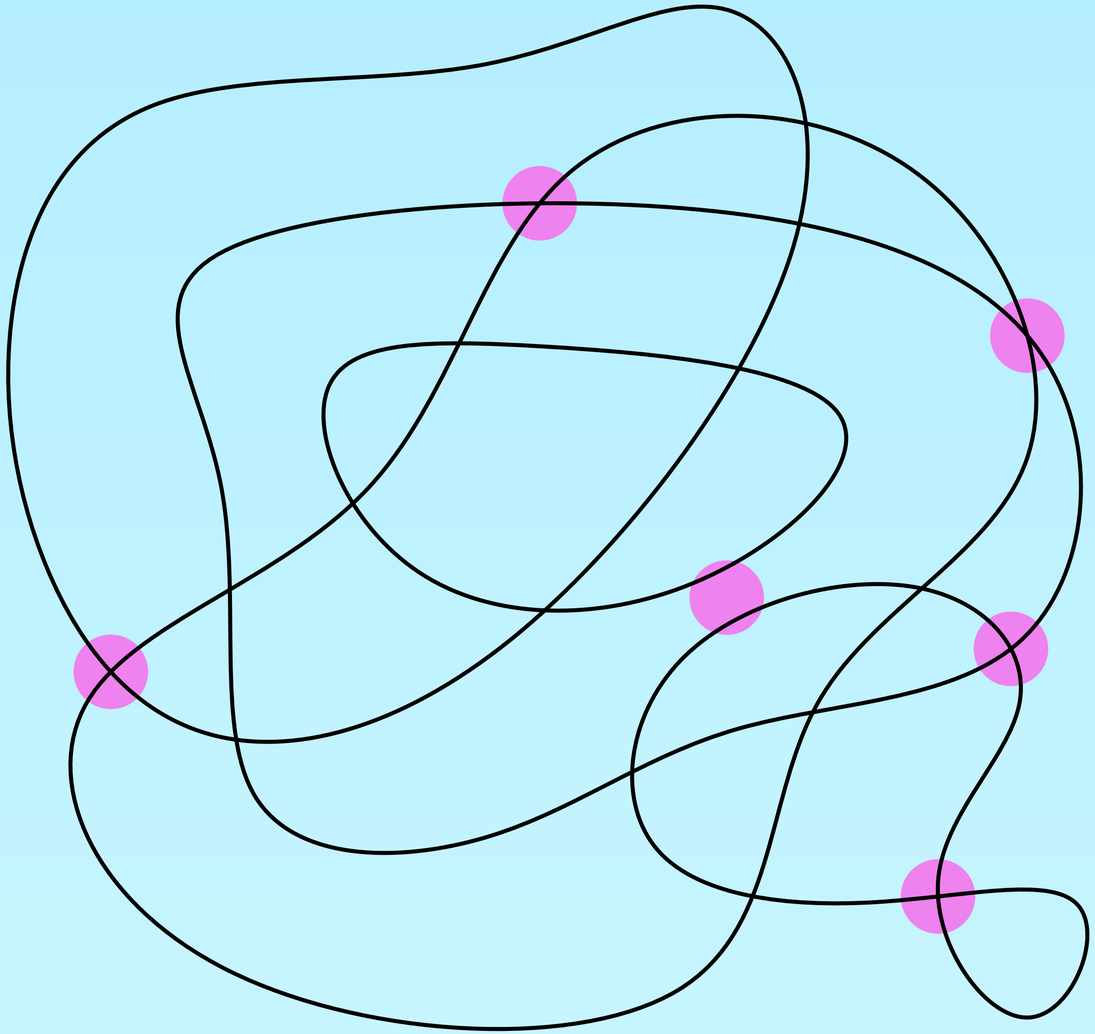
Problem statement.
Given a switchable curve arrangement, can we reconfigure the switches such that:

- all self-intersections are removed?



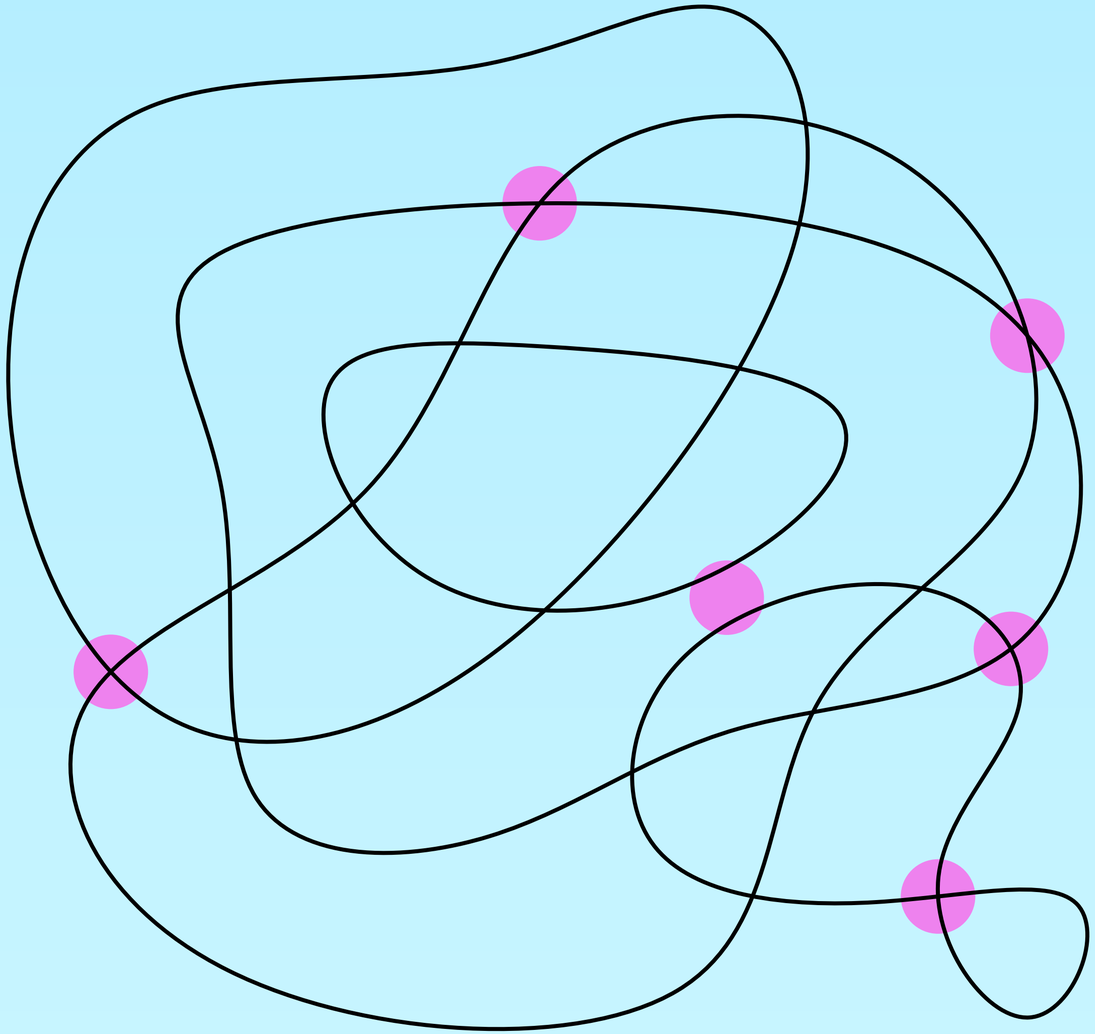
Problem statement.
Given a switchable curve arrangement, can we reconfigure the switches such that:

- all self-intersections are removed?
- all popular faces are removed?



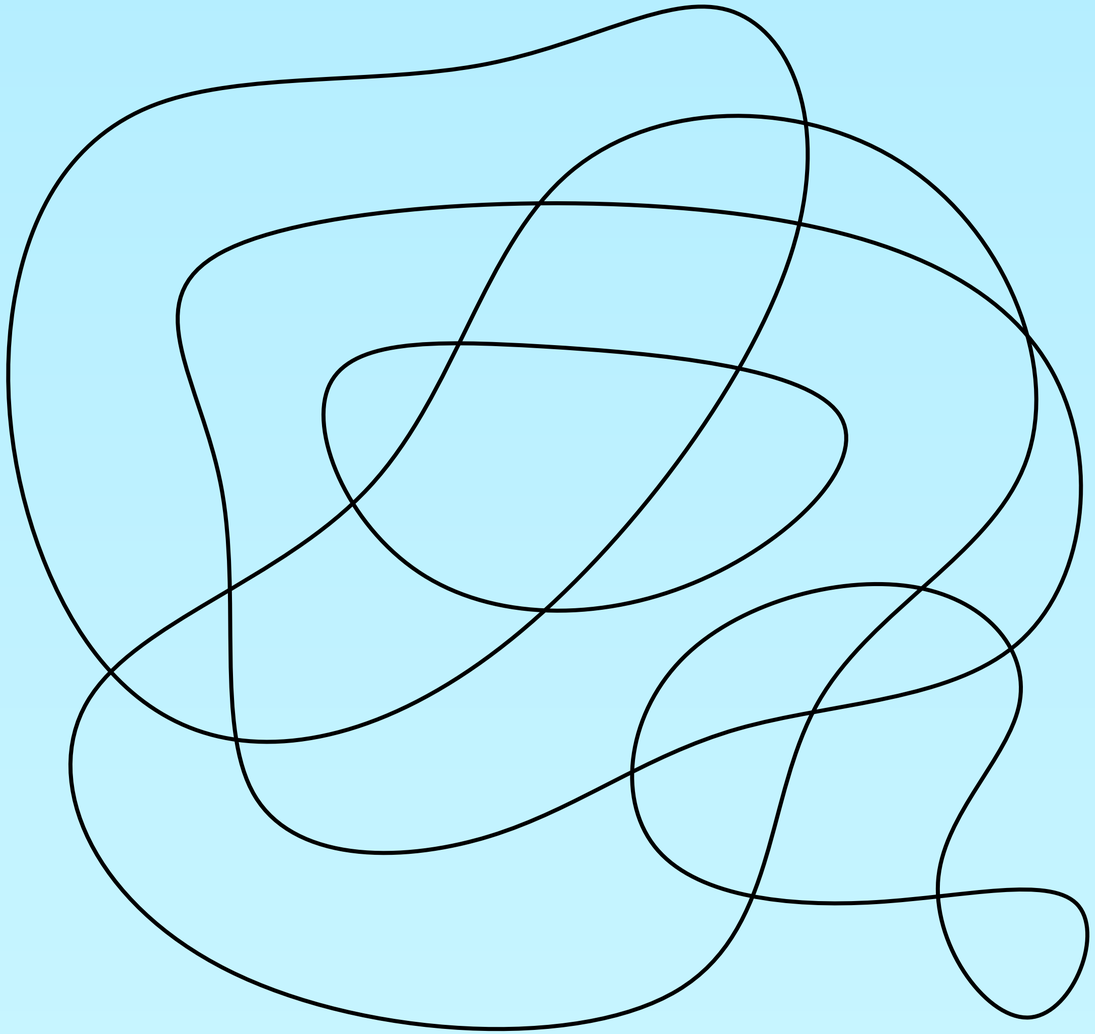
Problem statement.
Given a switchable curve arrangement, can we reconfigure the switches such that:

- all self-intersections are removed?
- all popular faces are removed?
- we minimize the number of switch operations used?

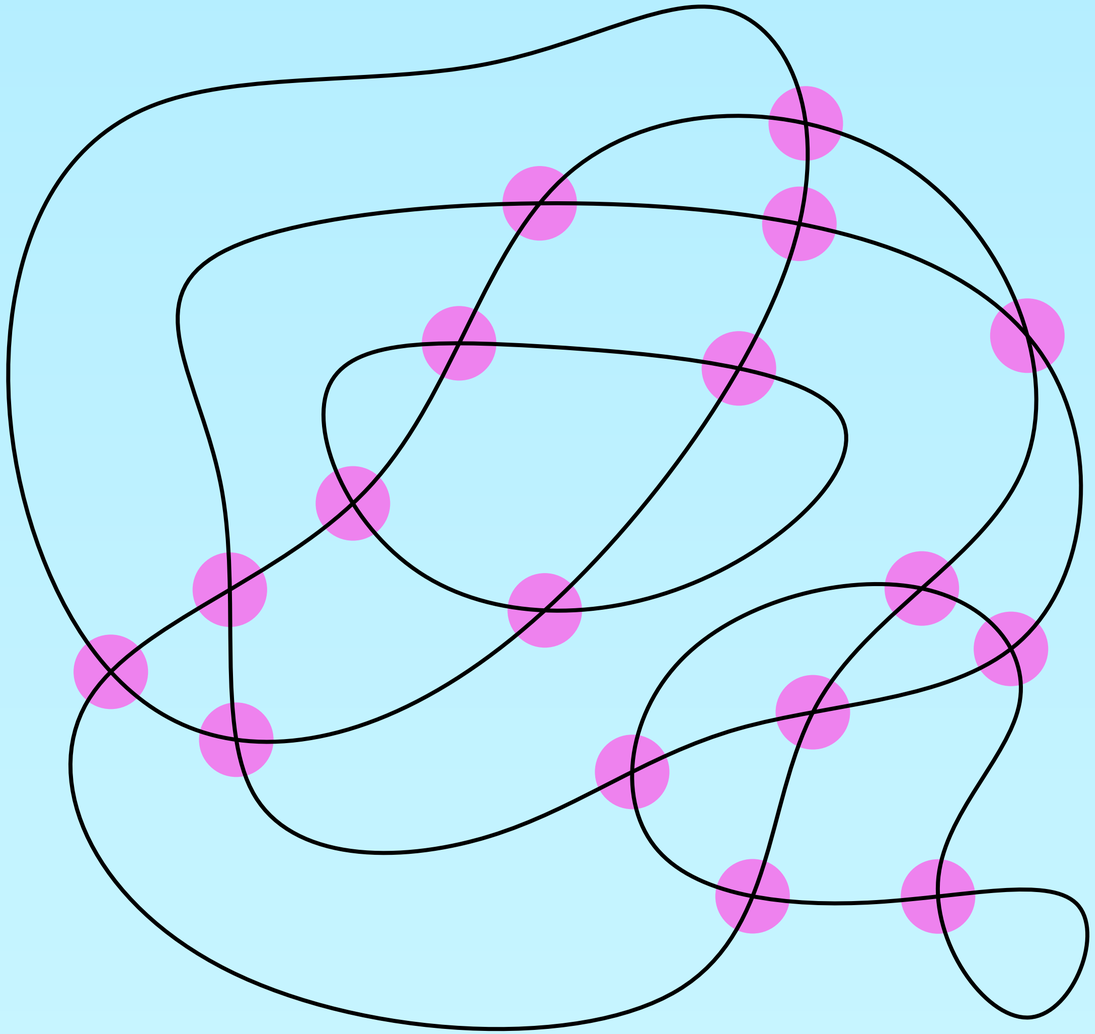


Problem statement.
Given a switchable curve arrangement, can we reconfigure the switches such that:

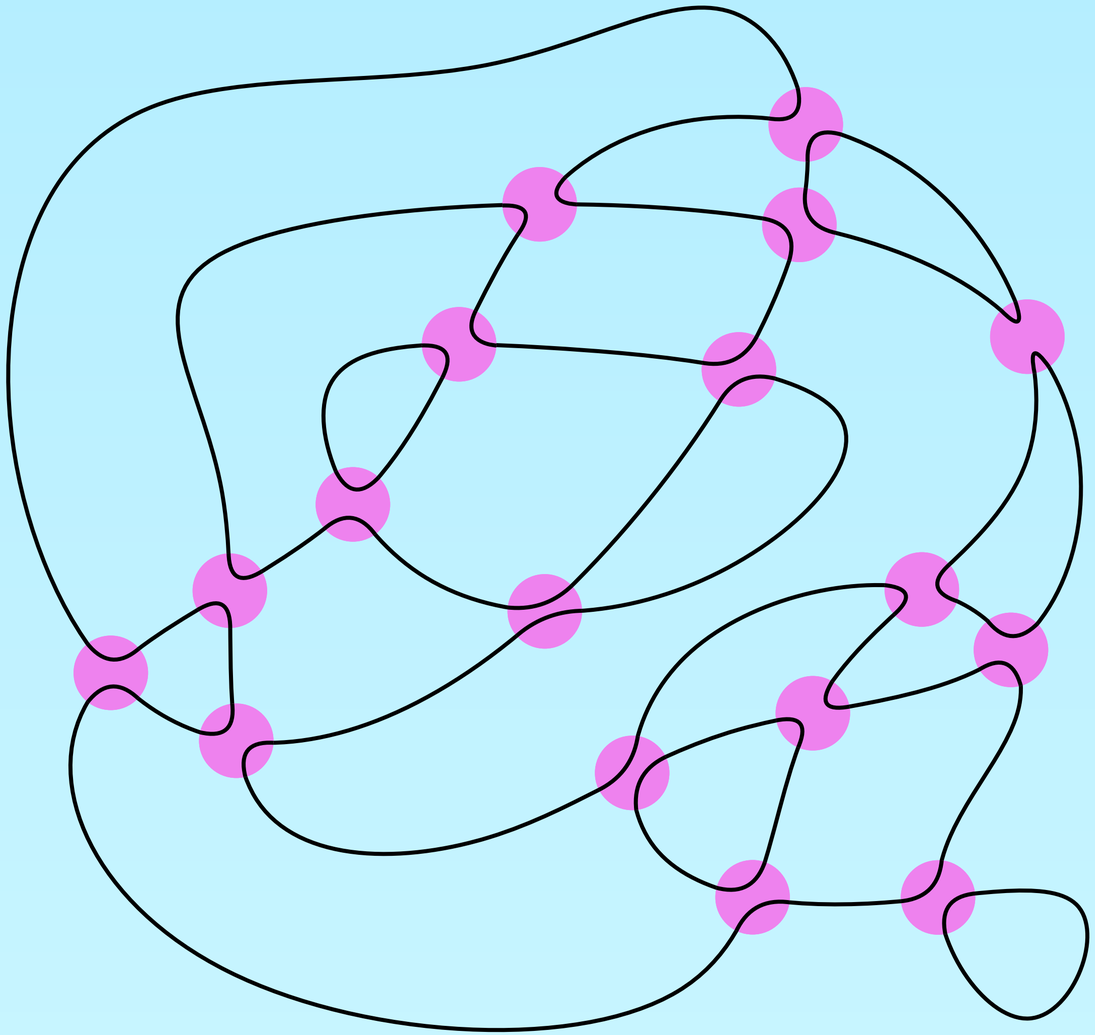
- all self-intersections are removed?
- all popular faces are removed?
- we minimize the number of switch operations used?
- we minimize the number of self-intersections or popular faces in the output?



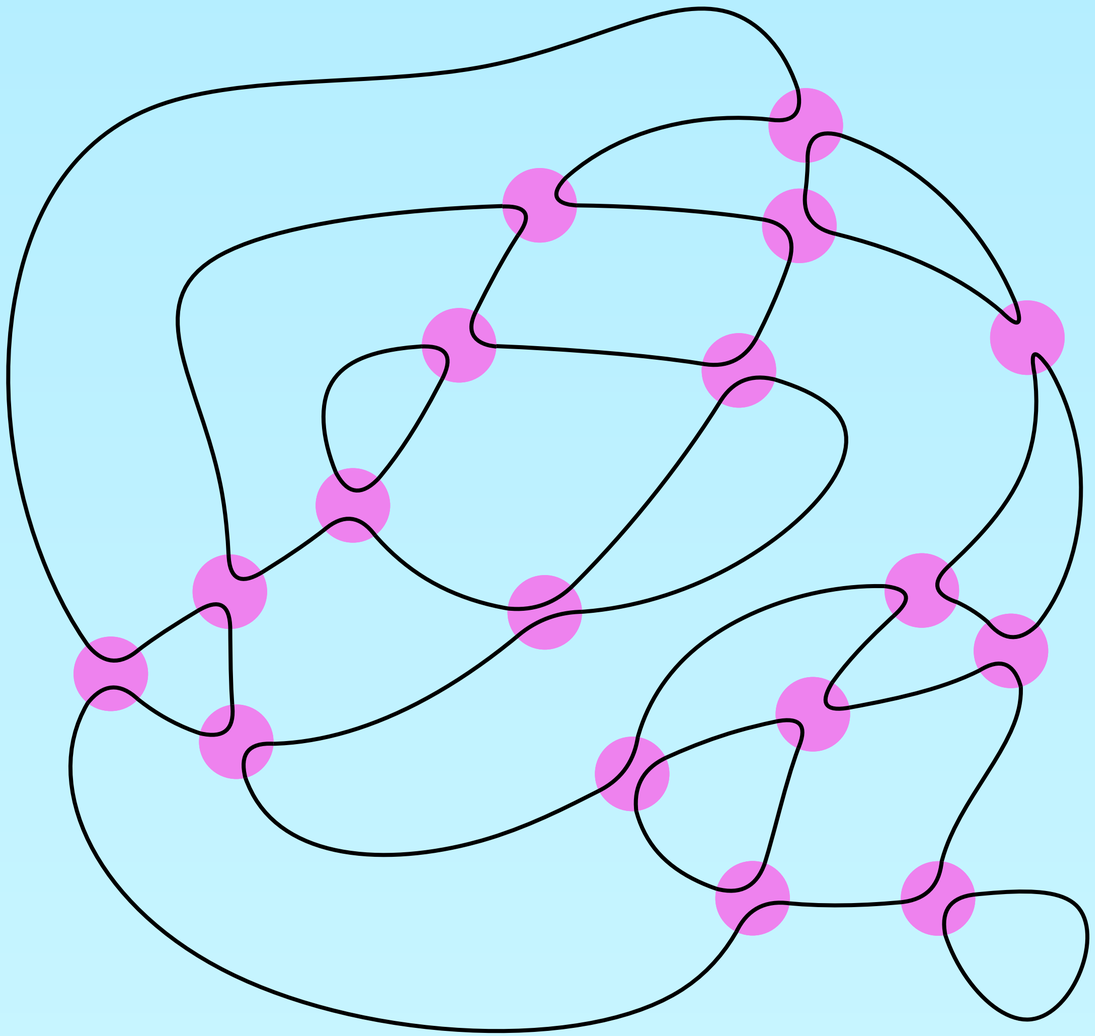
Observation.
When all vertices of the arrangement are switches, we can always remove all self-intersections and popular faces.



Observation.
When all vertices of the arrangement are switches, we can always remove all self-intersections and popular faces.



Observation.
When all vertices of the arrangement are switches, we can always remove all self-intersections and popular faces.



Observation.
When all vertices of the arrangement are switches, we can always remove all self-intersections and popular faces.

We probably want to allow as many switches as possible, but we want to switch as few of them as necessary.

Negative results:

Positive results:

Negative results:

Testing whether it is possible to remove either all self-intersections or all popular faces from a switchable arrangement is NP-hard.

Positive results:

Negative results:

Testing whether it is possible to remove either all self-intersections or all popular faces from a switchable arrangement is NP-hard.

Minimizing the number of switch operations is also NP-hard, even when all vertices are switches.

Positive results:

Negative results:

Testing whether it is possible to remove either all self-intersections or all popular faces from a switchable arrangement is NP-hard.

Minimizing the number of switch operations is also NP-hard, even when all vertices are switches.

Positive results:

When we have an arrangement with only k switches, removing all self-intersections of all popular faces is FPT in k.

Negative results:

Testing whether it is possible to remove either all self-intersections or all popular faces from a switchable arrangement is NP-hard.

Minimizing the number of switch operations is also NP-hard, even when all vertices are switches.

Positive results:

When we have an arrangement with only k switches, removing all self-intersections of all popular faces is FPT in k.

When we have k non-switchable crossings, removing all self-intersections is also FPT in k.

Negative results:

Testing whether it is possible to remove either all self-intersections or all popular faces from a switchable arrangement is NP-hard.

Minimizing the number of switch operations is also NP-hard, even when all vertices are switches.

Positive results:

When we have an arrangement with only k switches, removing all self-intersections of all popular faces is FPT in k.

When we have k non-switchable crossings, removing all self-intersections is also FPT in k.

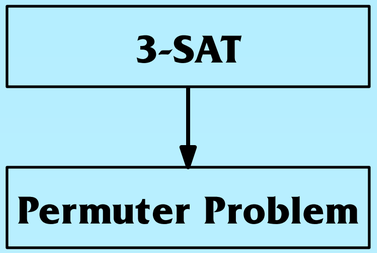
Still open for popular faces!

Part 3
HOW?

Hardness.

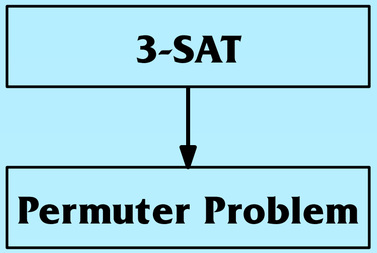
Hardness.

We have a series of reductions, all based on a common problem: the permuter problem.



Hardness.

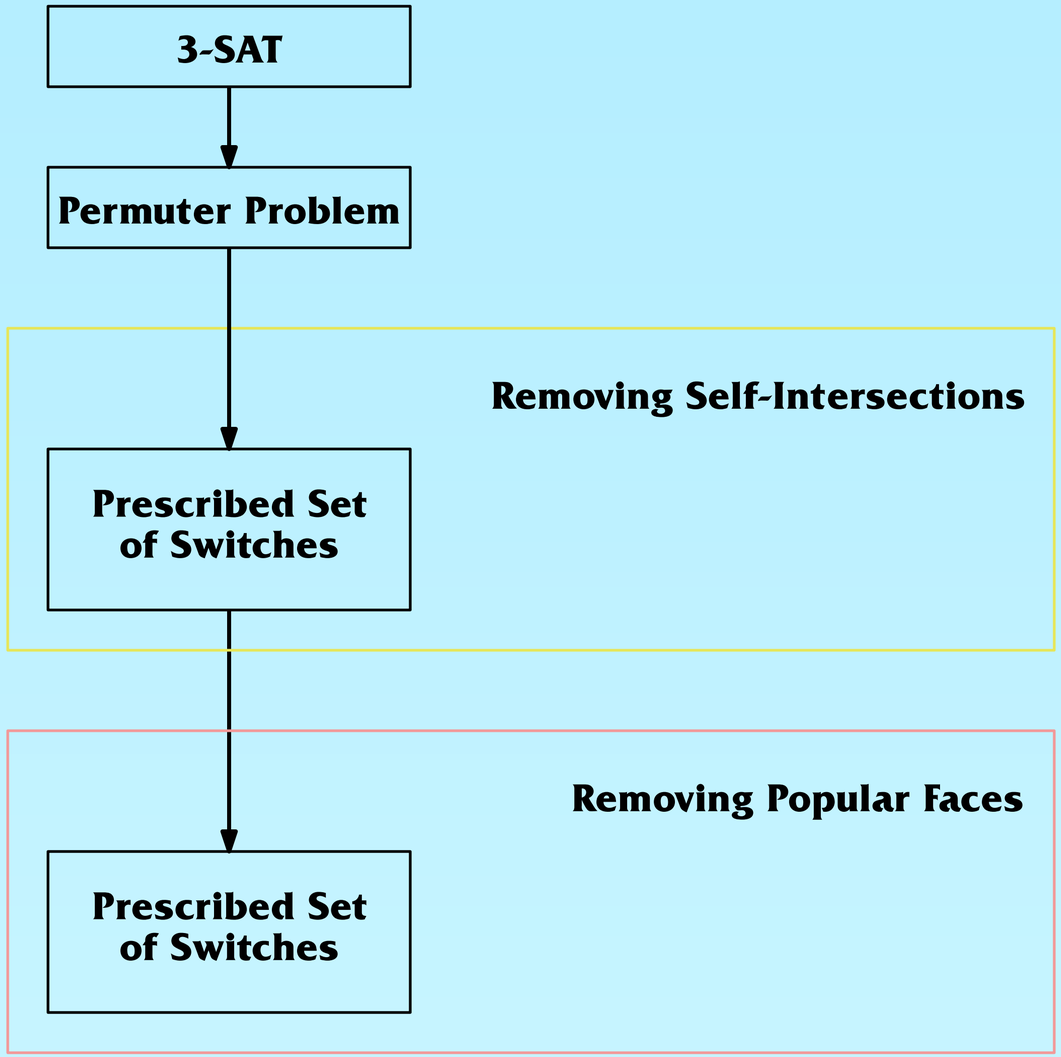
We have a series of reductions, all based on a common problem: the permuter problem.



Hardness.

We have a series of reductions, all based on a common problem: the permuter problem.

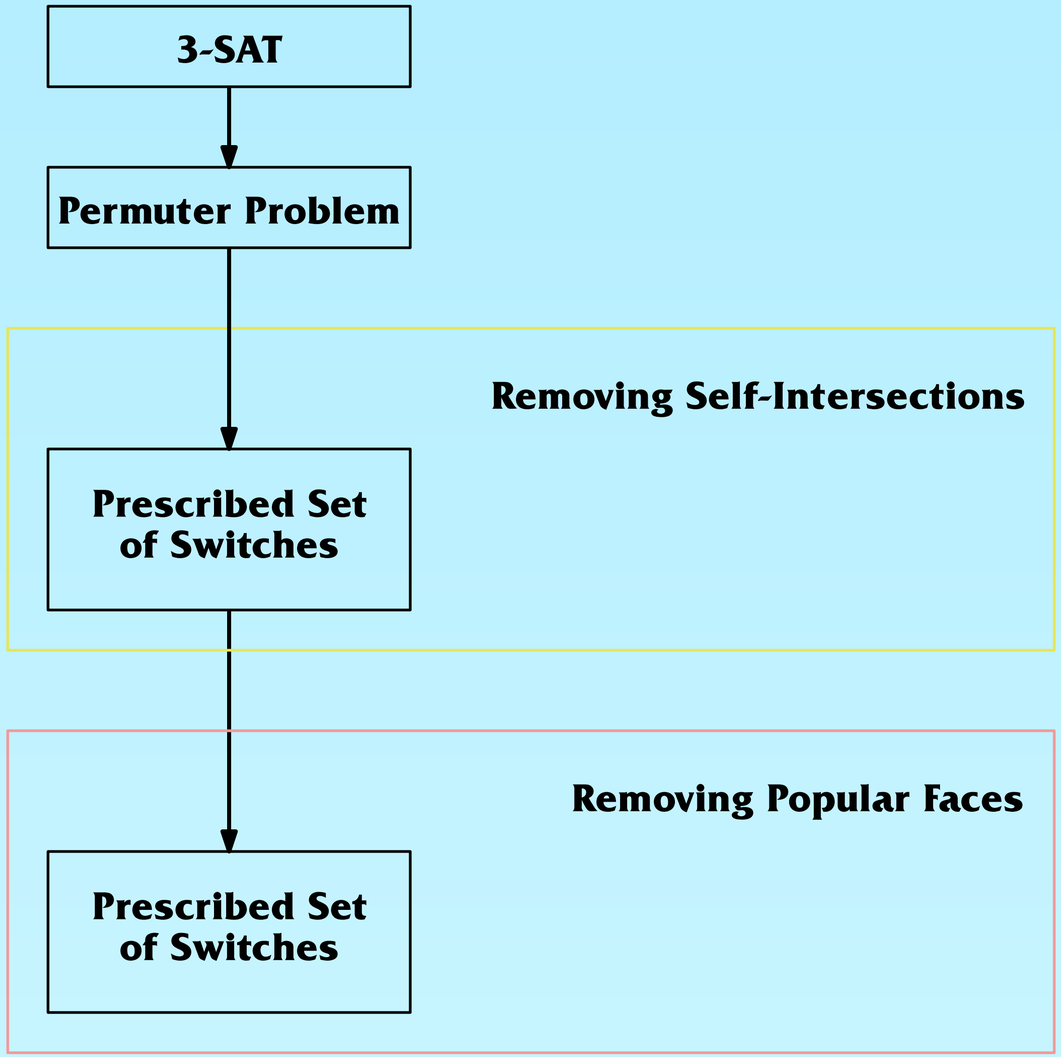
From there, we reduce first to removing self-intersections, and then to popular faces.



Hardness.

We have a series of reductions, all based on a common problem: the permuter problem.

From there, we reduce first to removing self-intersections, and then to popular faces.

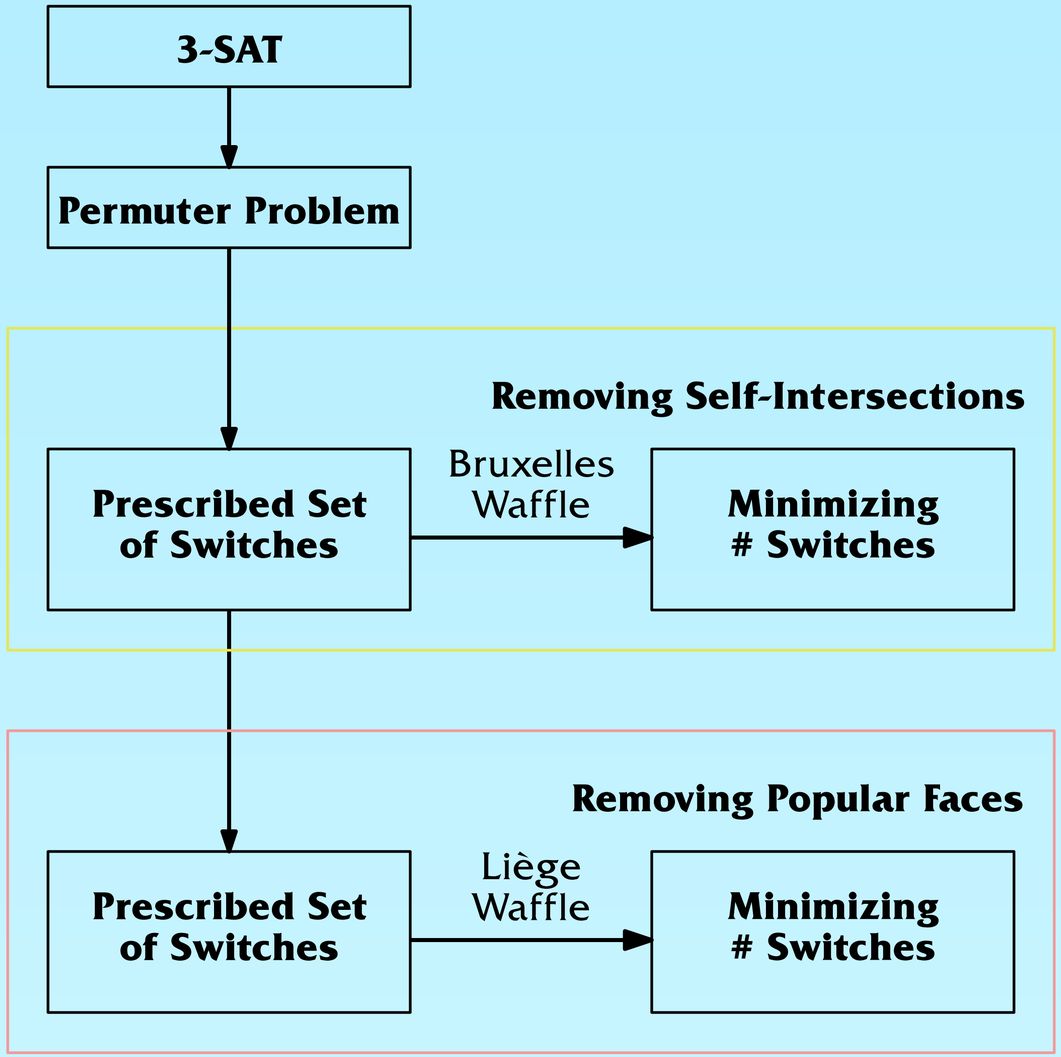


Hardness.

We have a series of reductions, all based on a common problem: the permuter problem.

From there, we reduce first to removing self-intersections, and then to popular faces.

Finally, we adapt the constructions to the case of minimizing switches.



Hardness.

We have a series of reductions, all based on a common problem: the permuter problem.

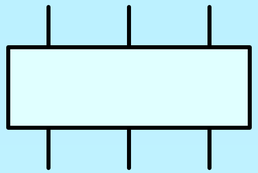
From there, we reduce first to removing self-intersections, and then to popular faces.

Finally, we adapt the constructions to the case of minimizing switches.

Permuter problem.

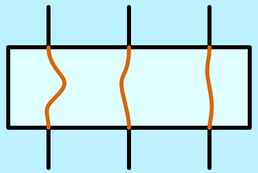
Permuter problem.

A permuter has k connectors on the top and k connectors on the bottom, that we can permute on the inside.



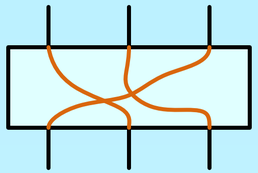
Permuter problem.

A permuter has k connectors on the top and k connectors on the bottom, that we can permute on the inside.



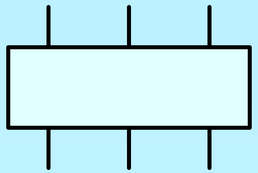
Permuter problem.

A permuter has k connectors on the top and k connectors on the bottom, that we can permute on the inside.



Permuter problem.

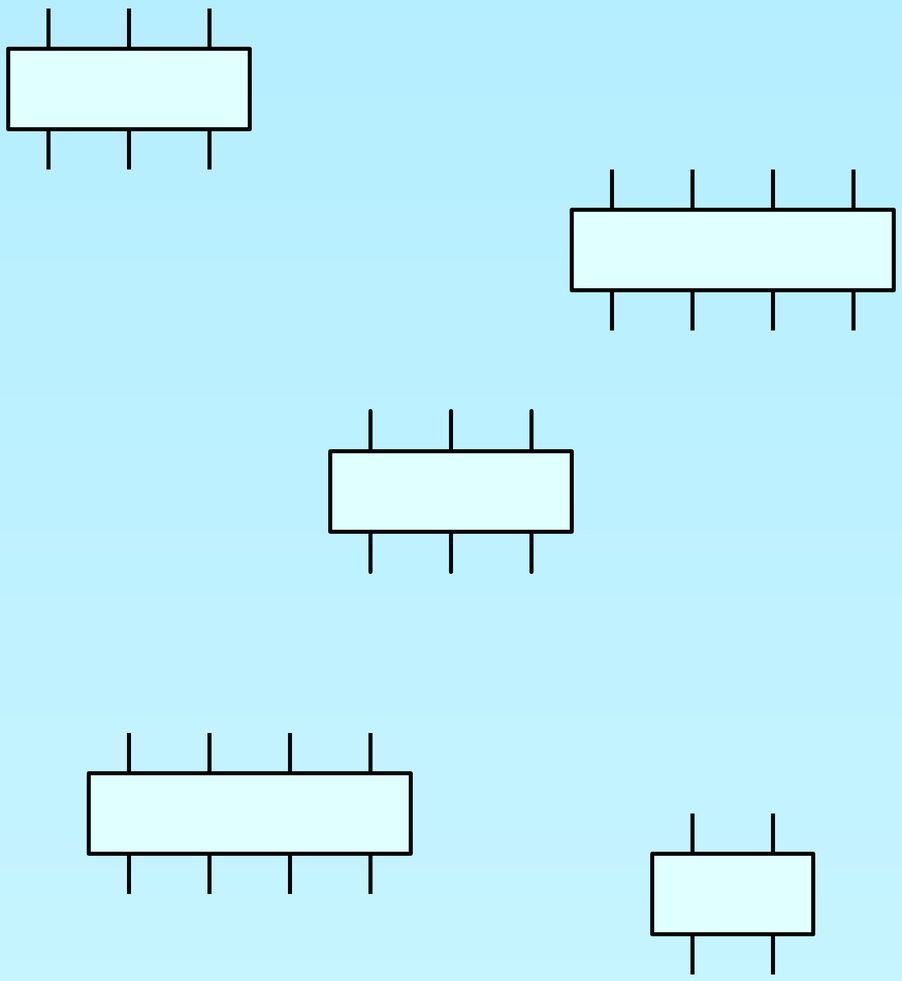
A permuter has k connectors on the top and k connectors on the bottom, that we can permute on the inside.



Permuter problem.

A permuter has k connectors on the top and k connectors on the bottom, that we can permute on the inside.

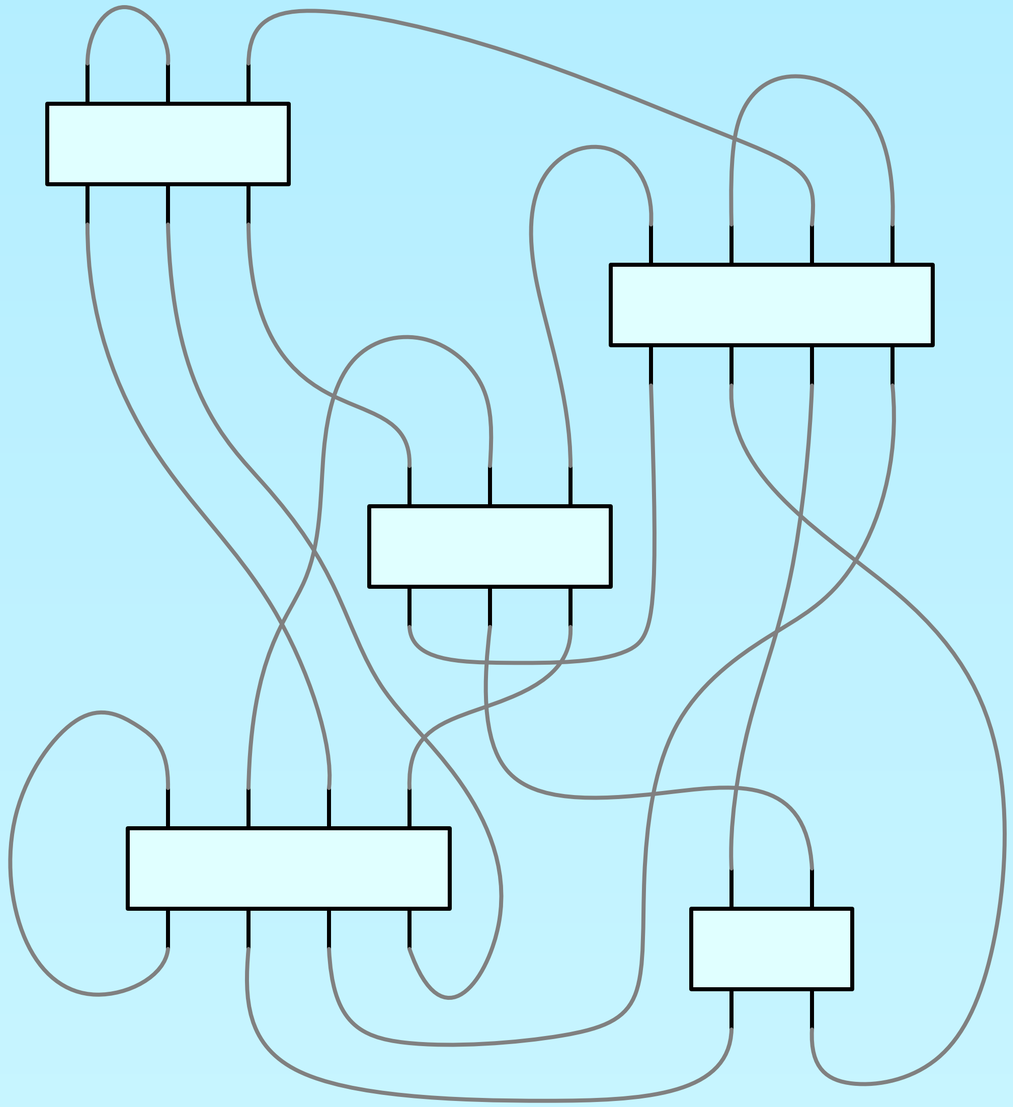
A permuter problem instance is a collection of permuters, whose connectors are linked with paths on the outside.



Permuter problem.

A permuter has k connectors on the top and k connectors on the bottom, that we can permute on the inside.

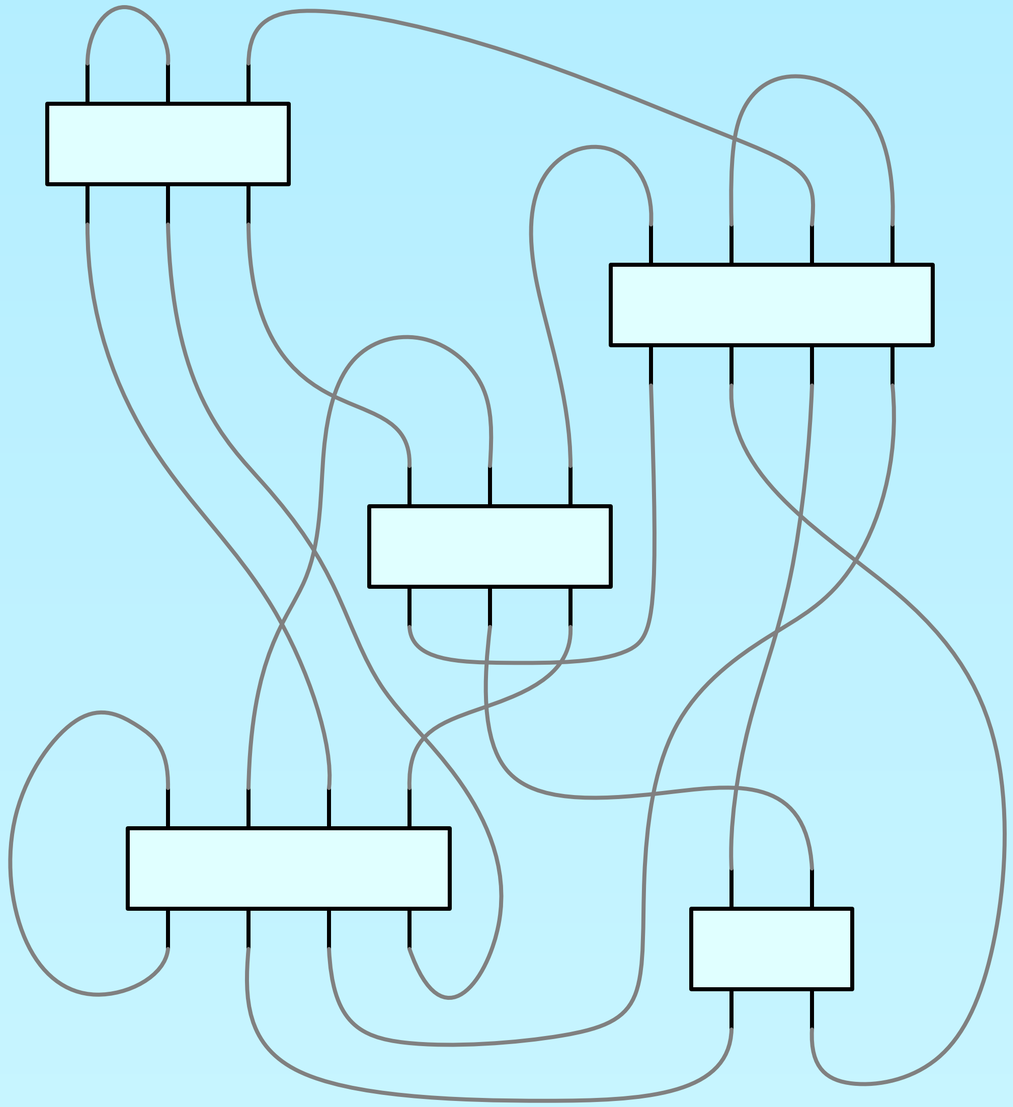
A permuter problem instance is a collection of permuters, whose connectors are linked with paths on the outside.



Permuter problem.

A permuter has k connectors on the top and k connectors on the bottom, that we can permute on the inside.

A permuter problem instance is a collection of permuters, whose connectors are linked with paths on the outside.

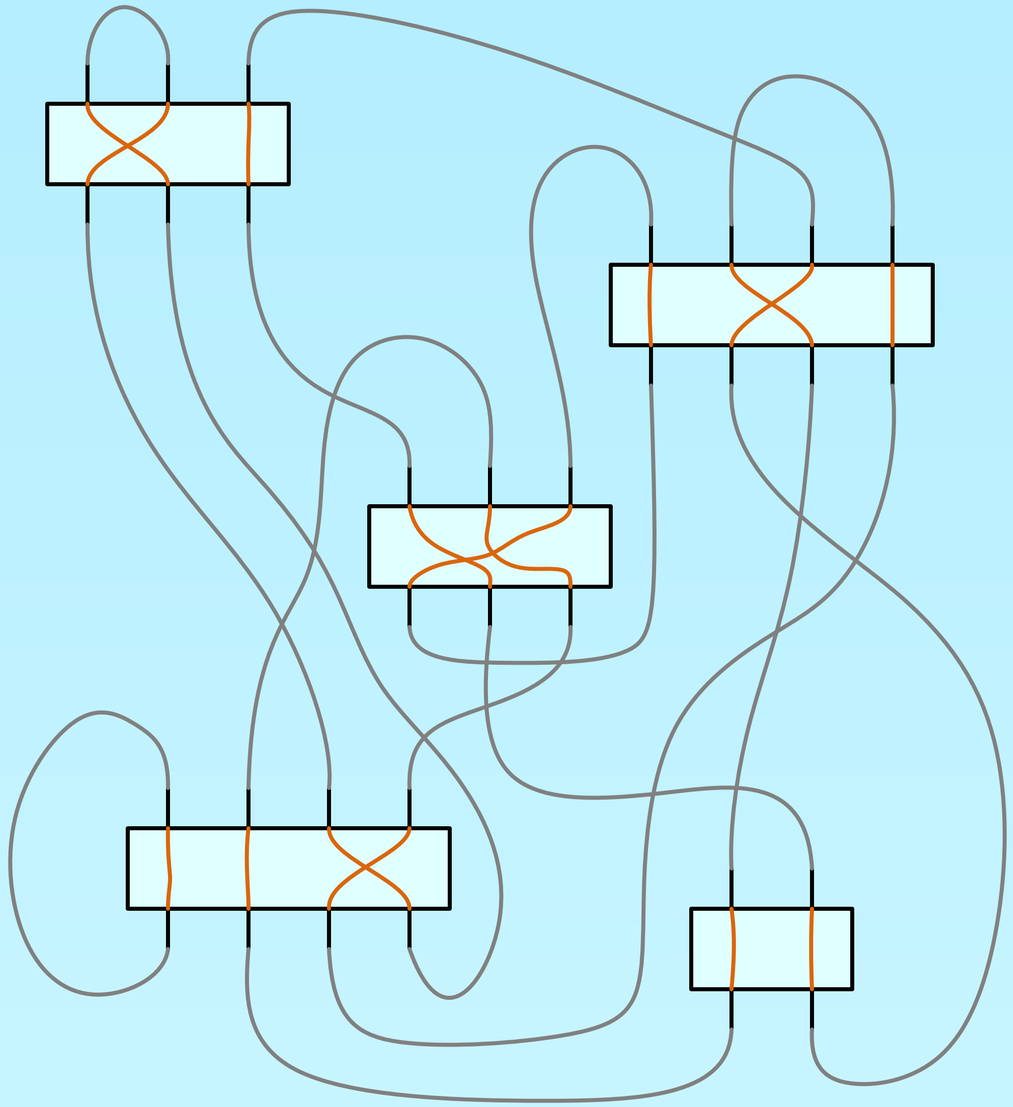


Permuter problem.

A permuter has k connectors on the top and k connectors on the bottom, that we can permute on the inside.

A permuter problem instance is a collection of permuters, whose connectors are linked with paths on the outside.

The output is a collection of permutations.

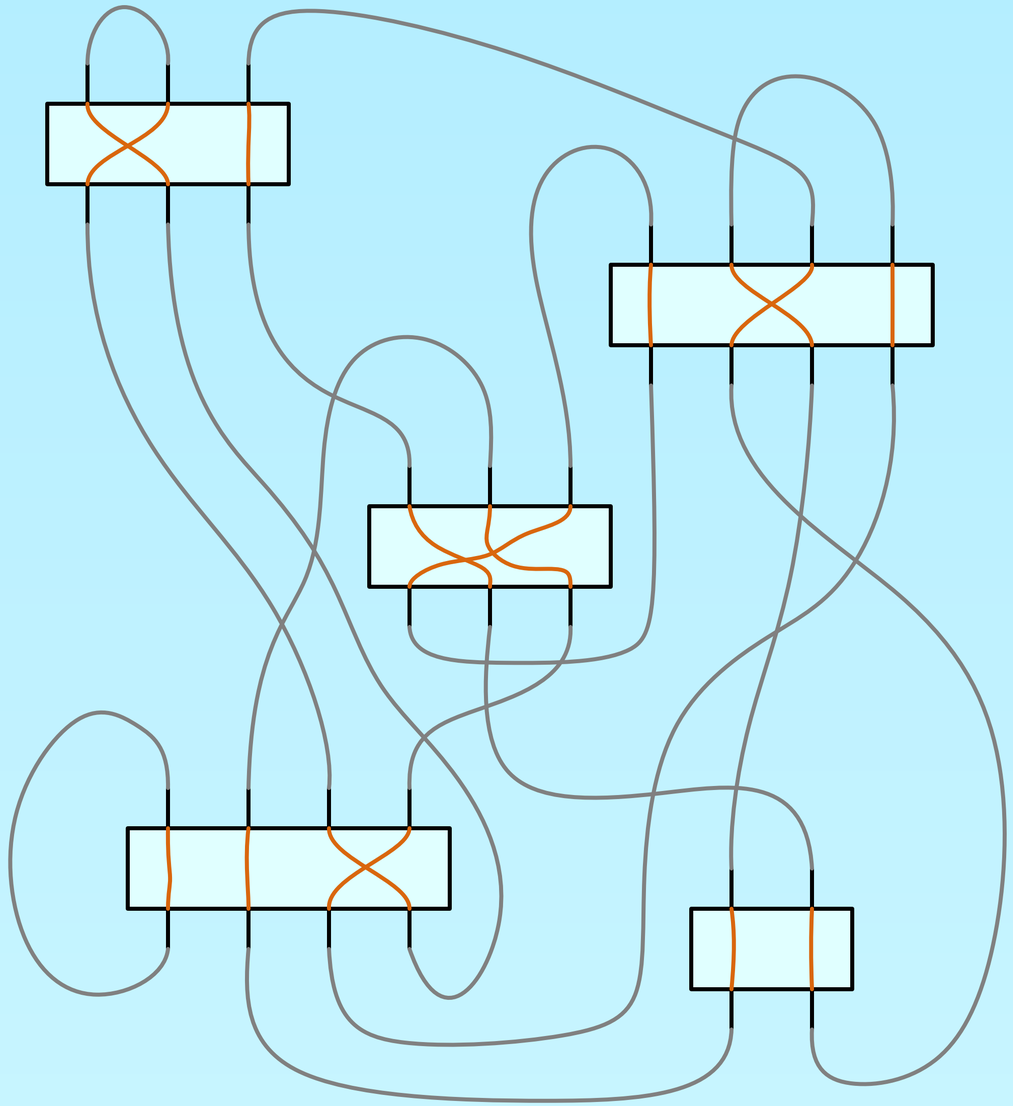


Permuter problem.

A permuter has k connectors on the top and k connectors on the bottom, that we can permute on the inside.

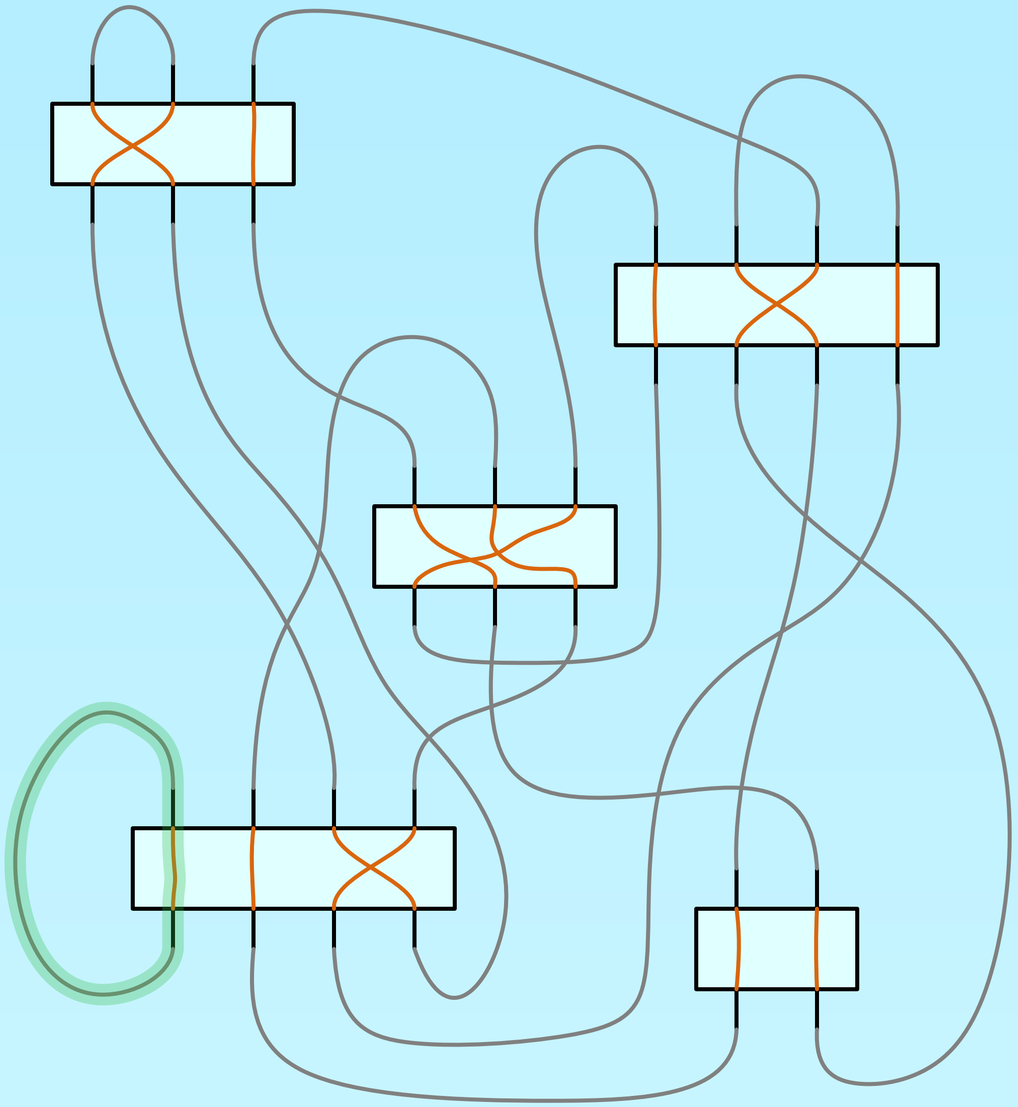
A permuter problem instance is a collection of permuters, whose connectors are linked with paths on the outside.

The output is a collection of permutations.



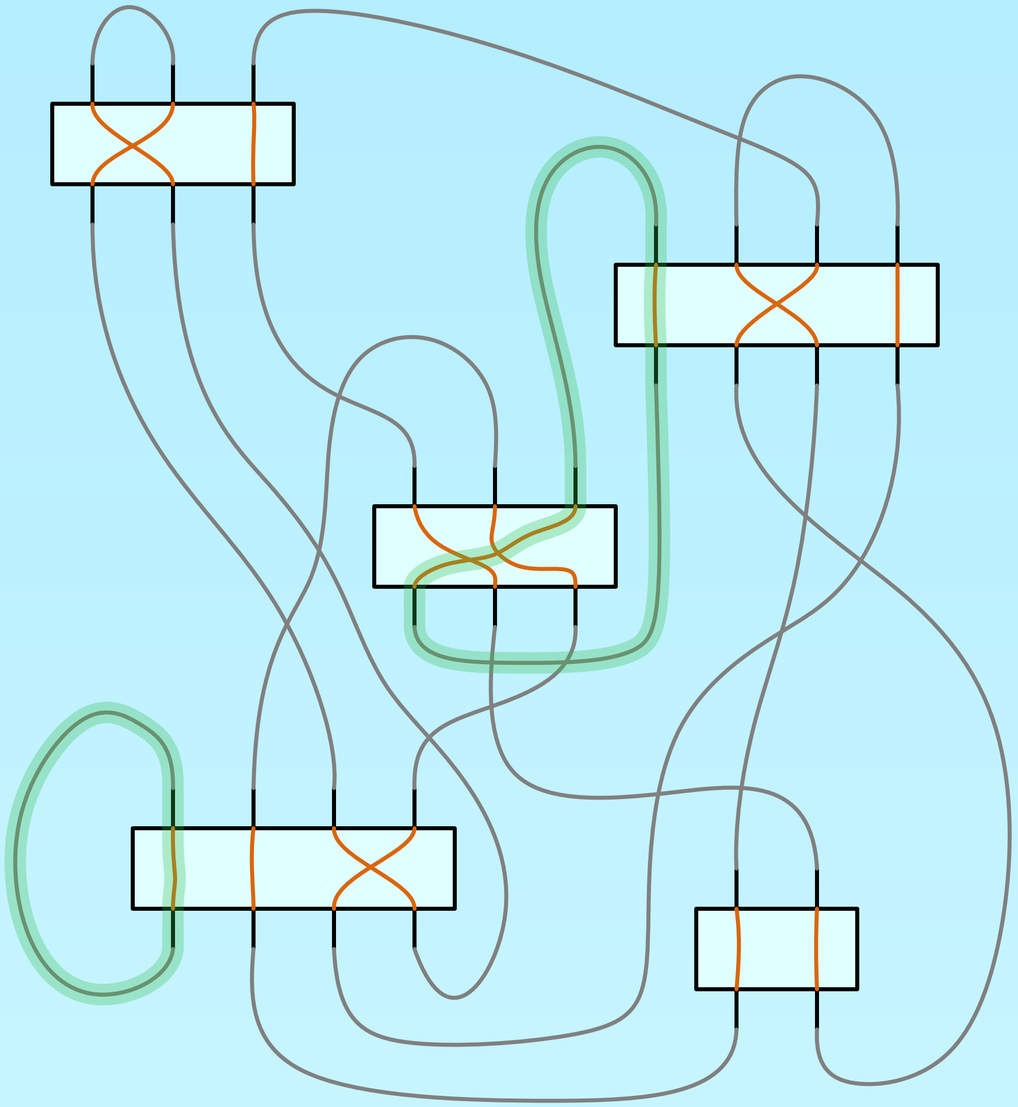
Permuter problem.

Goal: every resulting cycle should be simple.



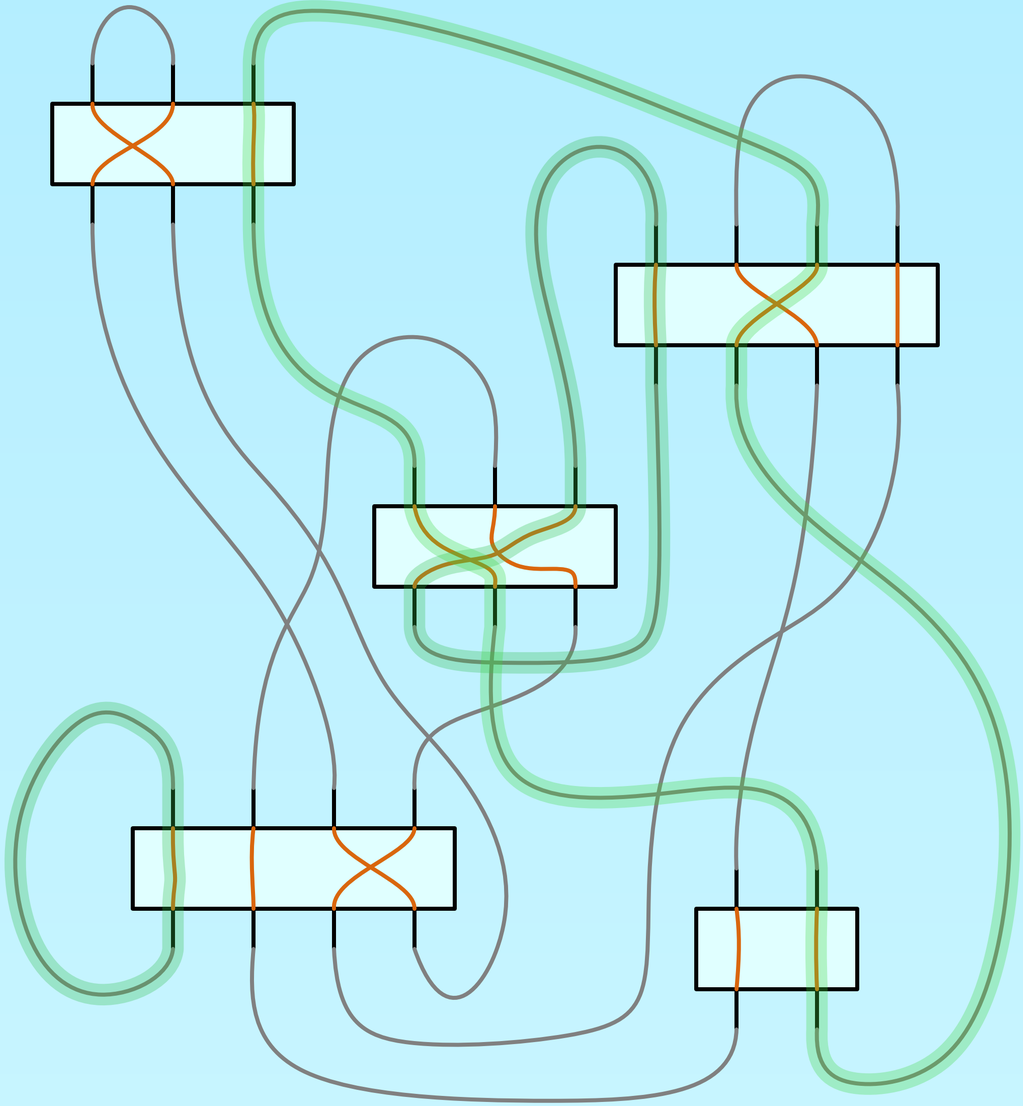
Permuter problem.

Goal: every resulting cycle should be simple.



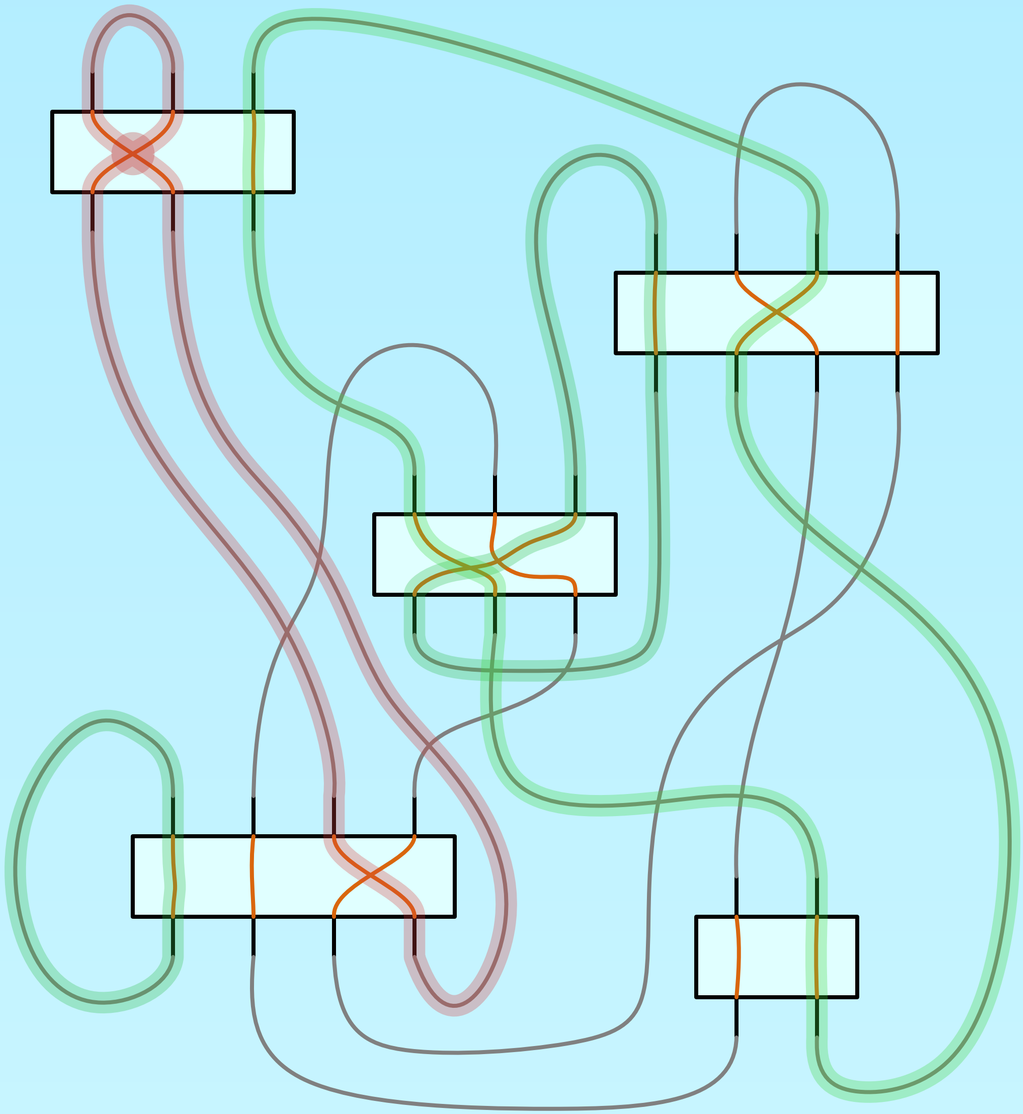
Permuter problem.

Goal: every resulting cycle should be simple.



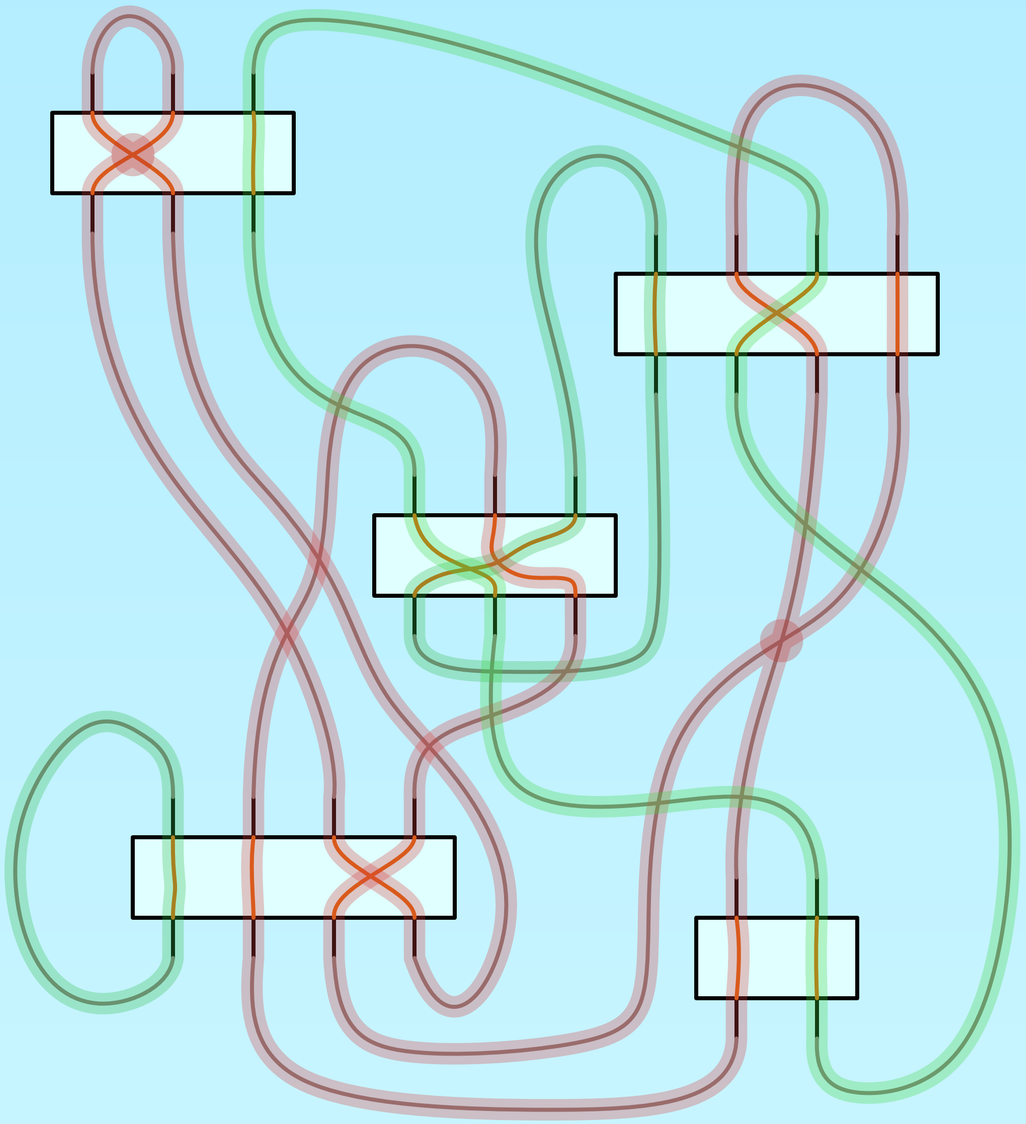
Permuter problem.

Goal: every resulting cycle should be simple.



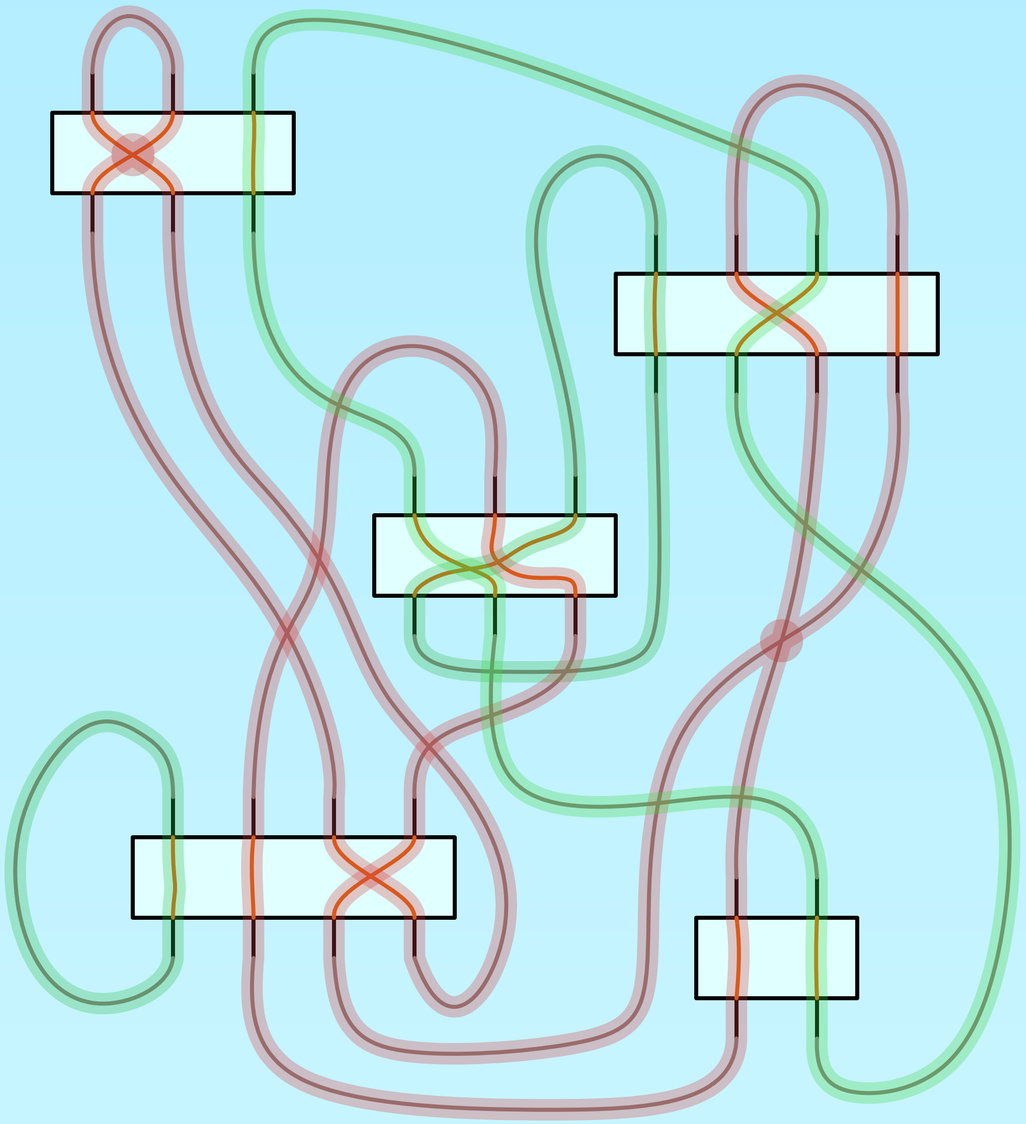
Permuter problem.

Goal: every resulting cycle should be simple.



Permuter problem.

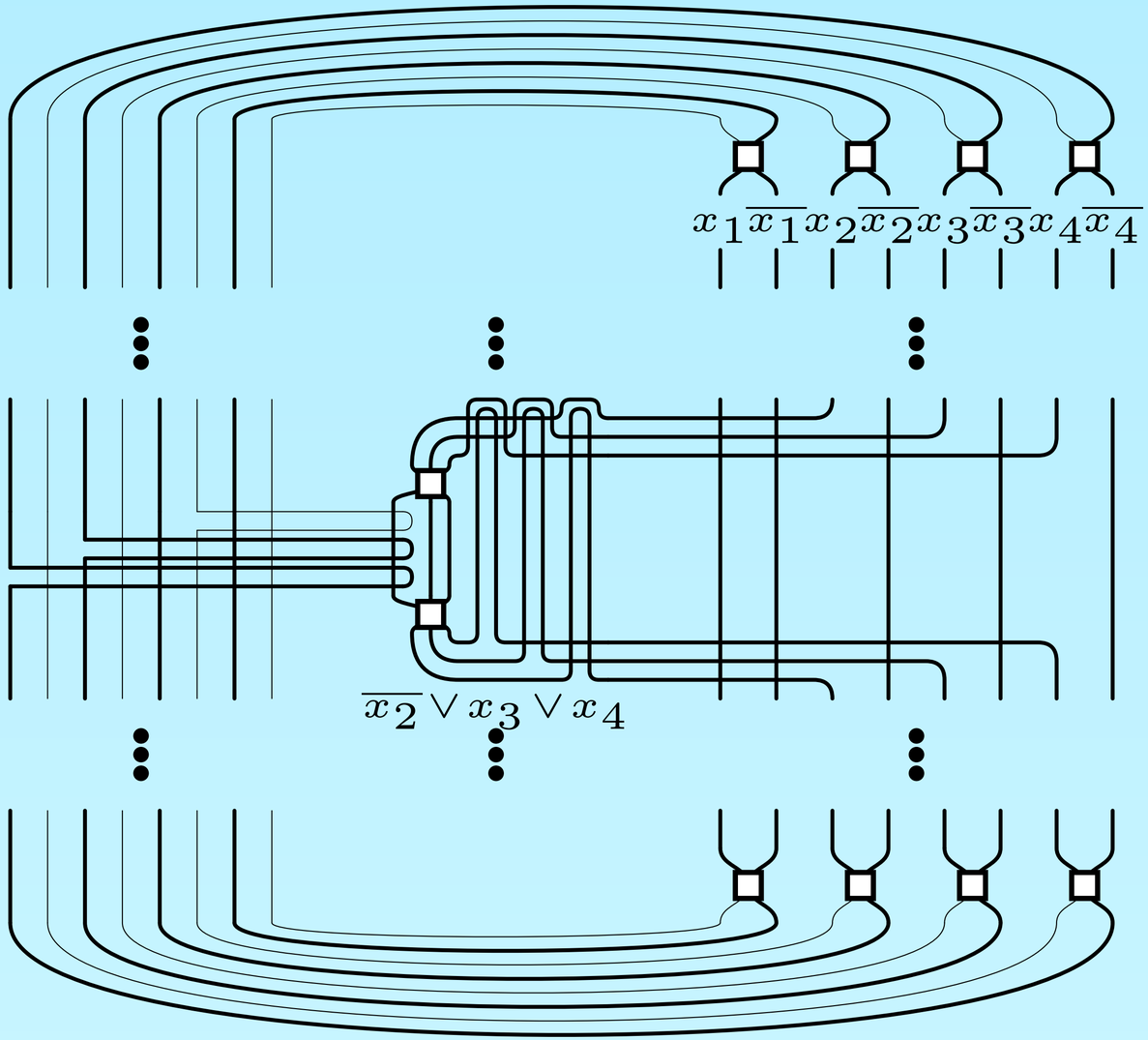
Goal: every resulting cycle should be simple.



Permuter problem.

Goal: every resulting cycle should be simple.

Claim: deciding whether a collection of permutations exists such that all cycles are simple, is NP-hard.



Permuter problem.

Goal: every resulting cycle should be simple.

Claim: deciding whether a collection of permutations exists such that all cycles are simple, is NP-hard.

Permuter problem to removing self-intersections.

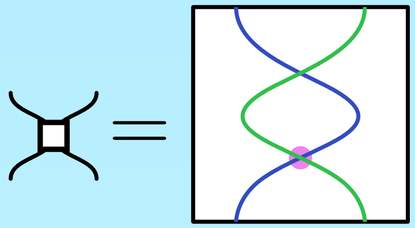
Permuter problem to removing self-intersections.

Outside the permuter gadgets, the contruction is identical.

Permuter problem to removing self-intersections.

Outside the permuter gadgets, the contruction is identical.

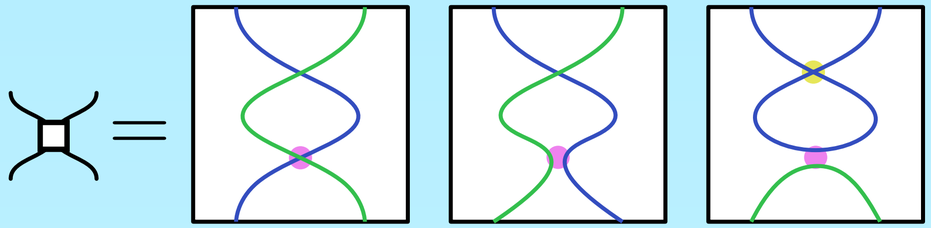
We can build a 2-permuter gadget using two "doubly-coiled" curves with a single switch.



Permuter problem to removing self-intersections.

Outside the permuter gadgets, the contruction is identical.

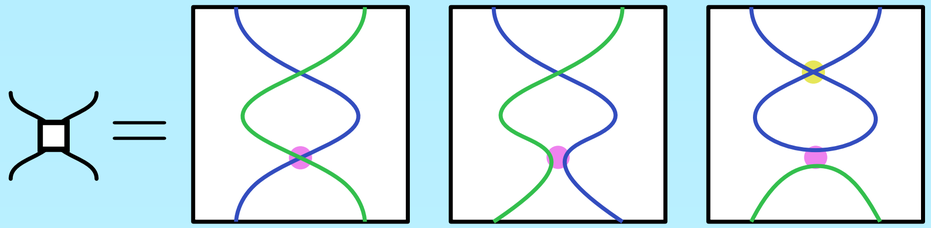
We can build a 2-permuter gadget using two "doubly-coiled" curves with a single switch.



Permuter problem to removing self-intersections.

Outside the permuter gadgets, the contruction is identical.

We can build a 2-permuter gadget using two "doubly-coiled" curves with a single switch.

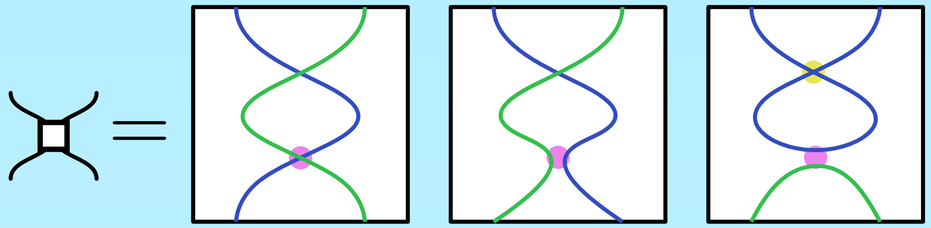


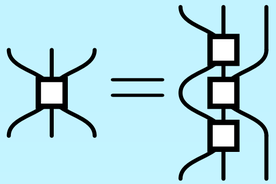
Permuter problem to removing self-intersections.

Outside the permuter gadgets, the contruction is identical.

We can build a 2-permuter gadget using two "doubly-coiled" curves with a single switch.

We can build a 3-permuter out of 3 2-permuters.





Permuter problem to removing self-intersections.

Outside the permuter gadgets, the contruction is identical.

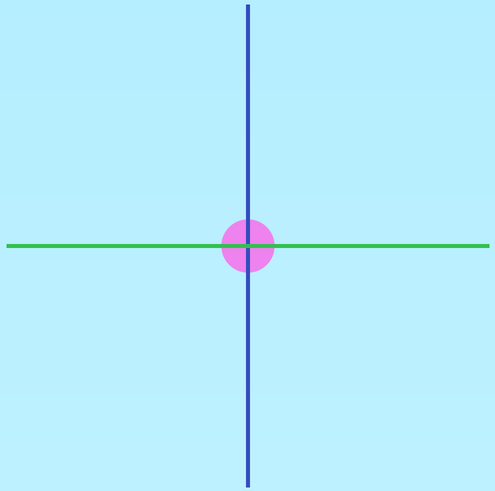
We can build a 2-permuter gadget using two "doubly-coiled" curves with a single switch.

We can build a 3-permuter out of 3 2-permuters.

Minimizing switches.

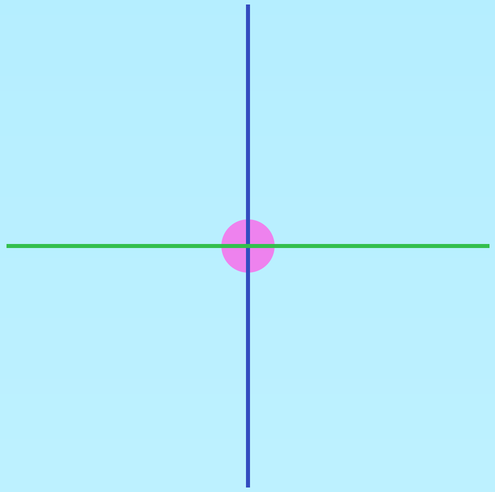
Minimizing switches.

Need a way to force switches to never be switched.



Minimizing switches.

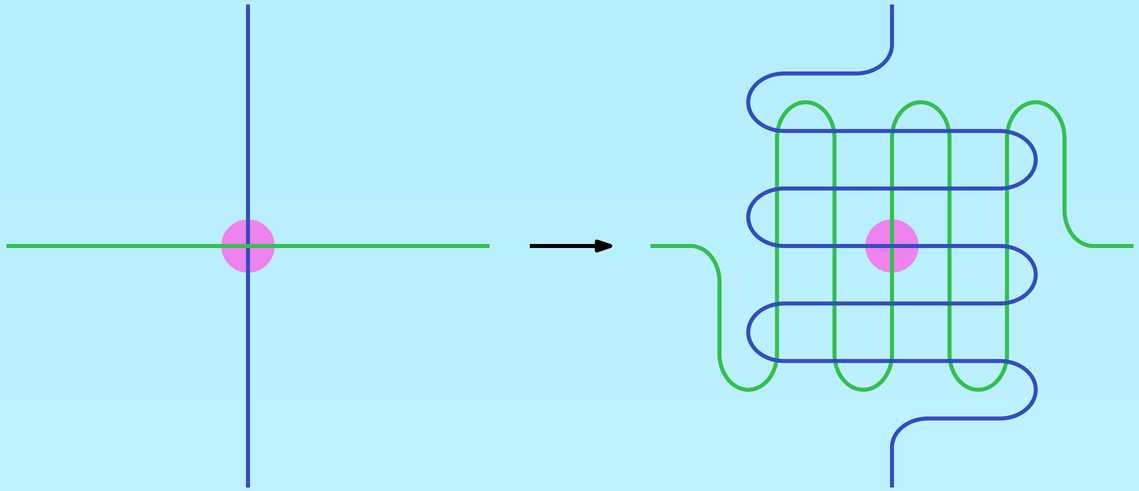
Need a way to force switches to never be switched.



Minimizing switches.

Need a way to force switches to never be switched.

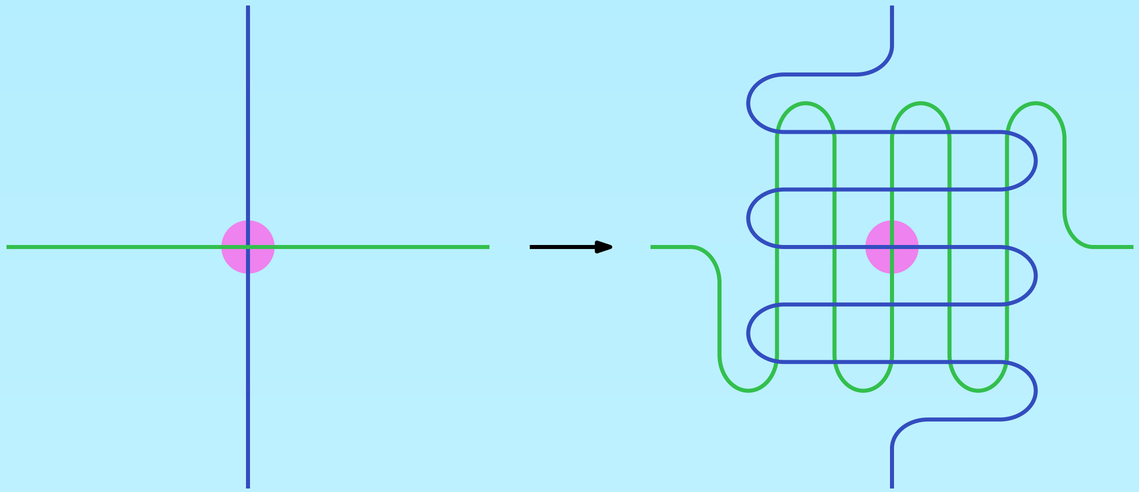
Introduce the Bruxelles waffle.



Minimizing switches.

Need a way to force switches to never be switched.

Introduce the Bruxelles waffle.

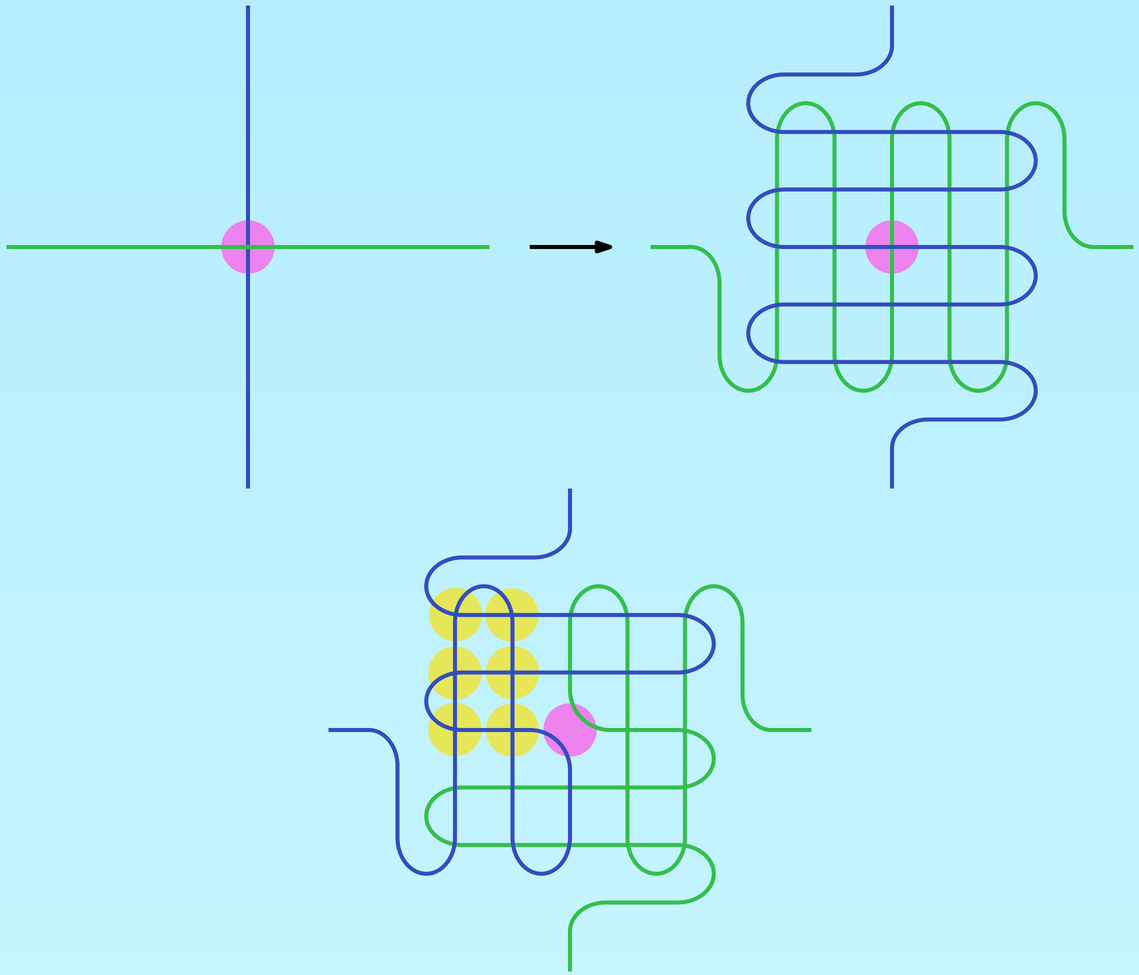


Minimizing switches.

Need a way to force switches to never be switched.

Introduce the Bruxelles waffle.

Lemma. In a Bruxelles waffle, any sequence of fewer than c switches must result in an arrangement with either the same combinatorial structure as the original, or at least one self-intersection.

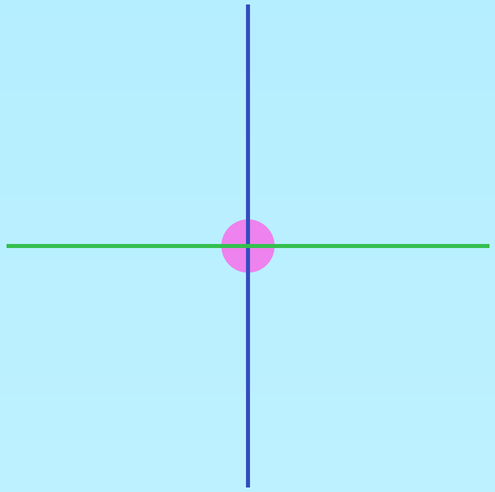


Minimizing switches.

Need a way to force switches to never be switched.

Introduce the Bruxelles waffle.

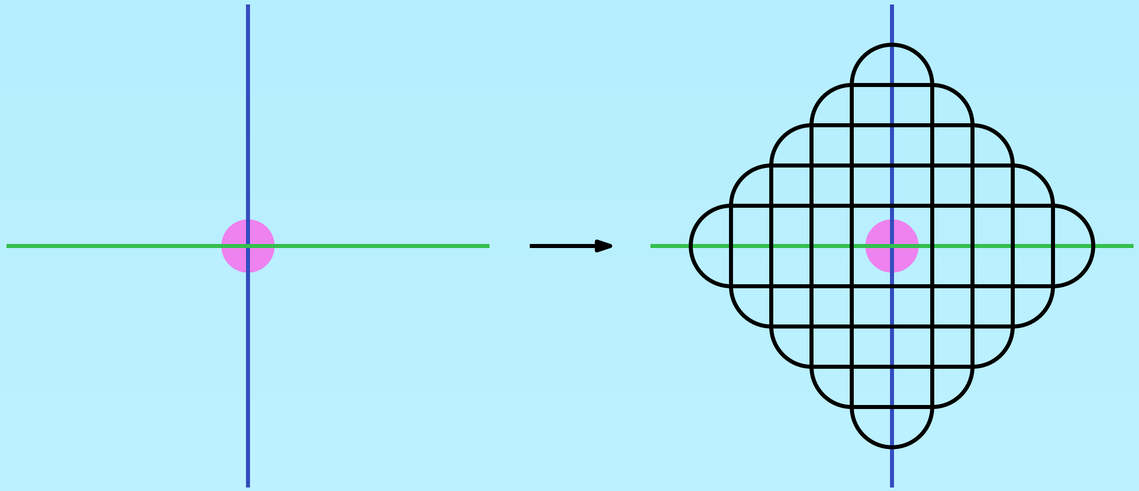
Lemma. In a Bruxelles waffle, any sequence of fewer than c switches must result in an arrangement with either the same combinatorial structure as the original, or at least one self-intersection.



Minimizing switches.

Need a way to force switches to never be switched.

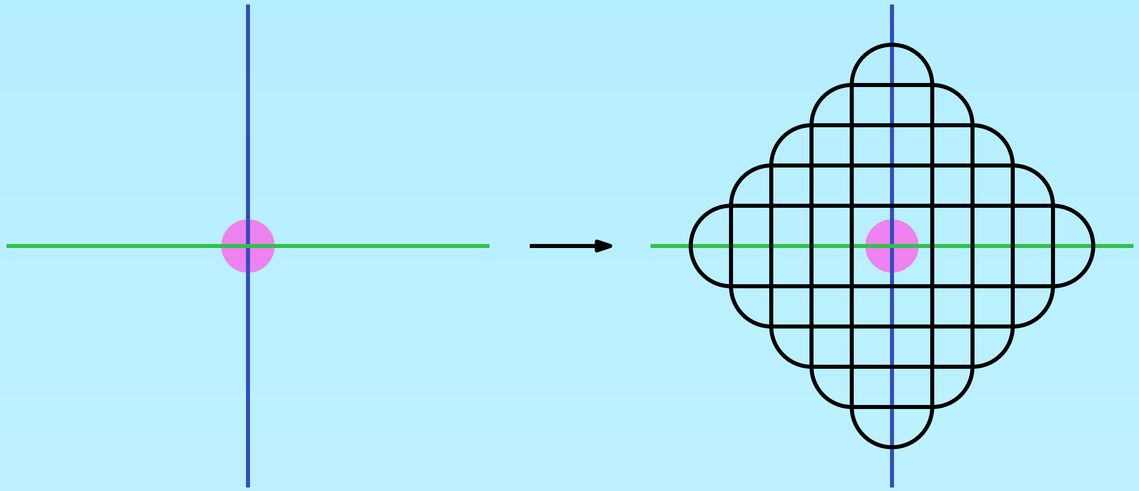
Introduce the Liège waffle.



Minimizing switches.

Need a way to force switches to never be switched.

Introduce the Liège waffle.

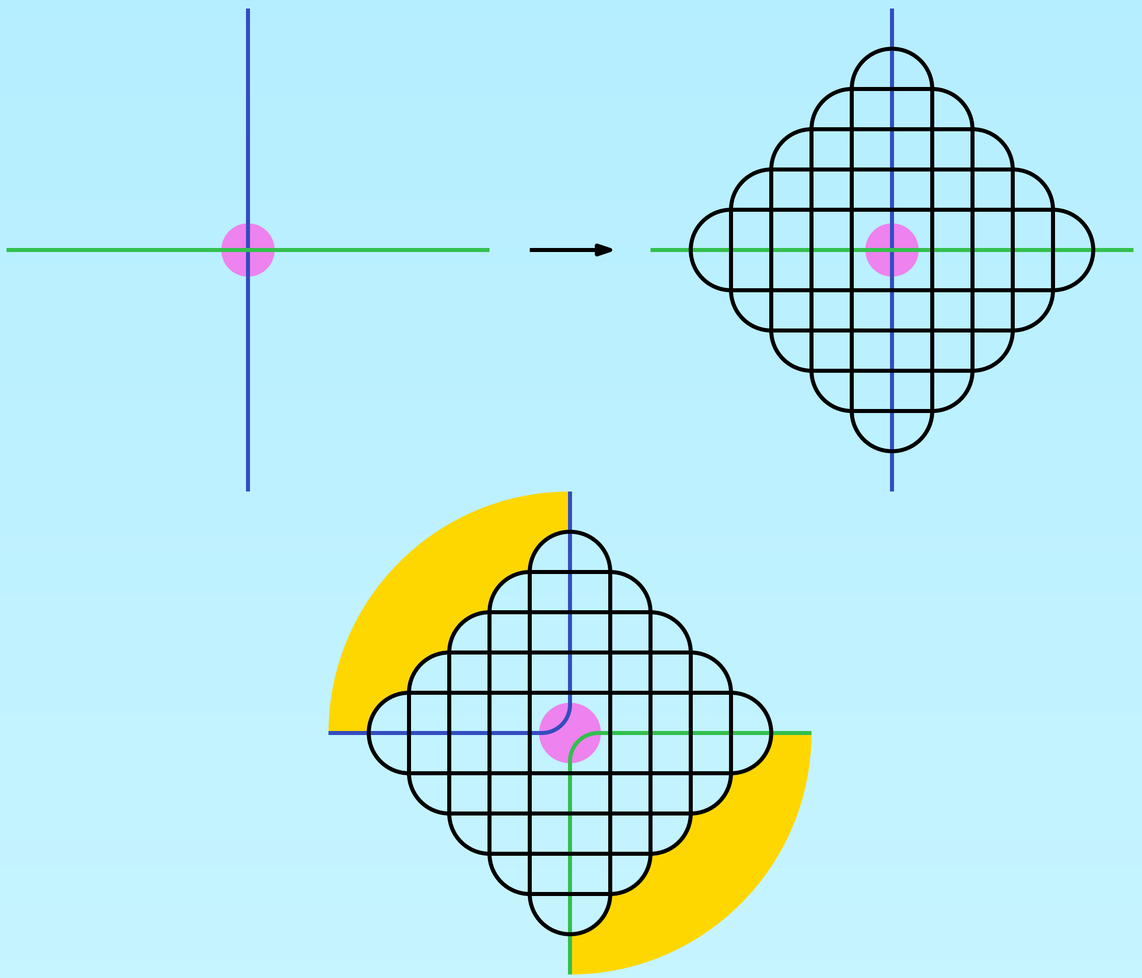


Minimizing switches.

Need a way to force switches to never be switched.

Introduce the Liège waffle.

Lemma. In a Liège waffle, any sequence of fewer than c switches must result in an arrangement with either the same combinatorial structure as the original, or at least one popular face.



Minimizing switches.

Need a way to force switches to never be switched.

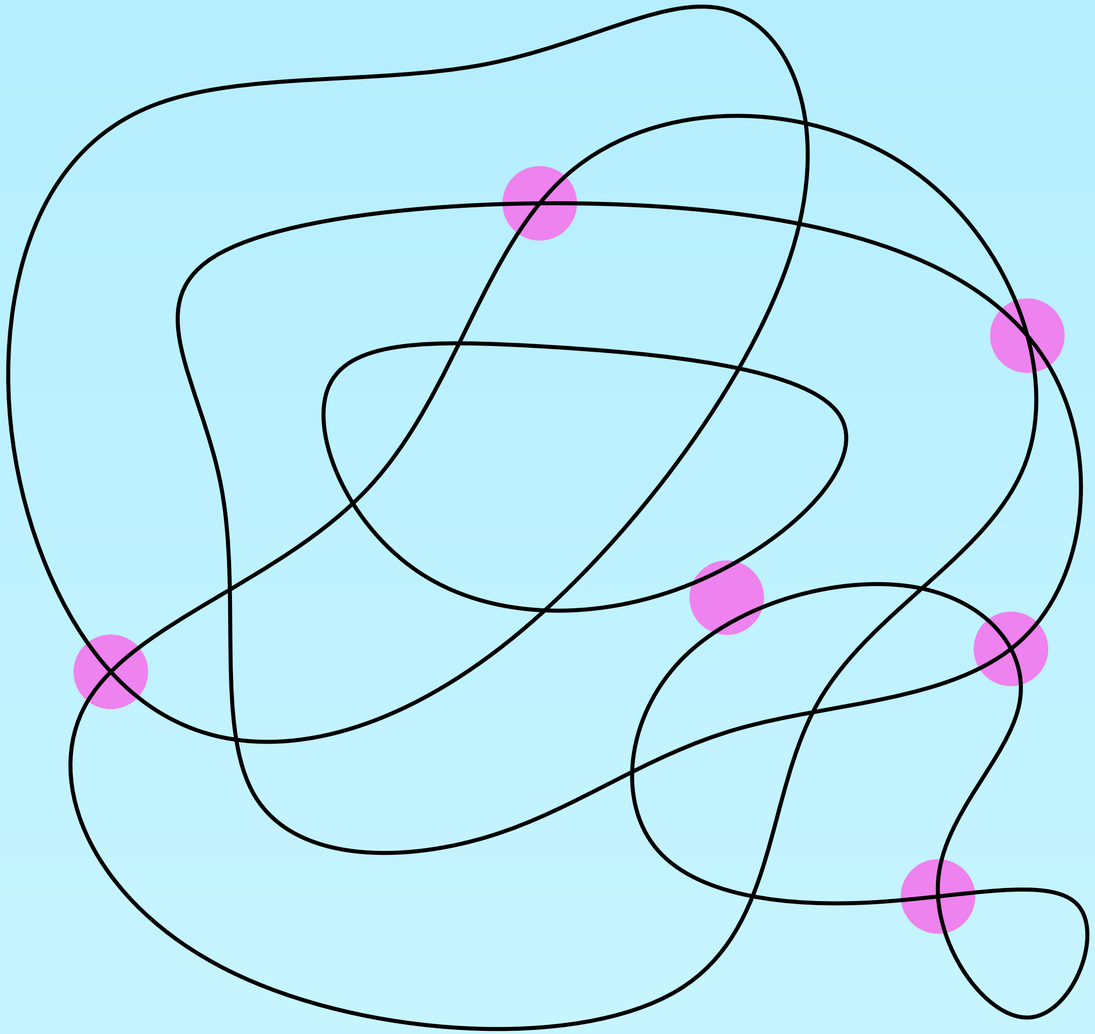
Introduce the Liège waffle.

Lemma. In a Liège waffle, any sequence of fewer than c switches must result in an arrangement with either the same combinatorial structure as the original, or at least one popular face.

Fixed Parameter Tractability.

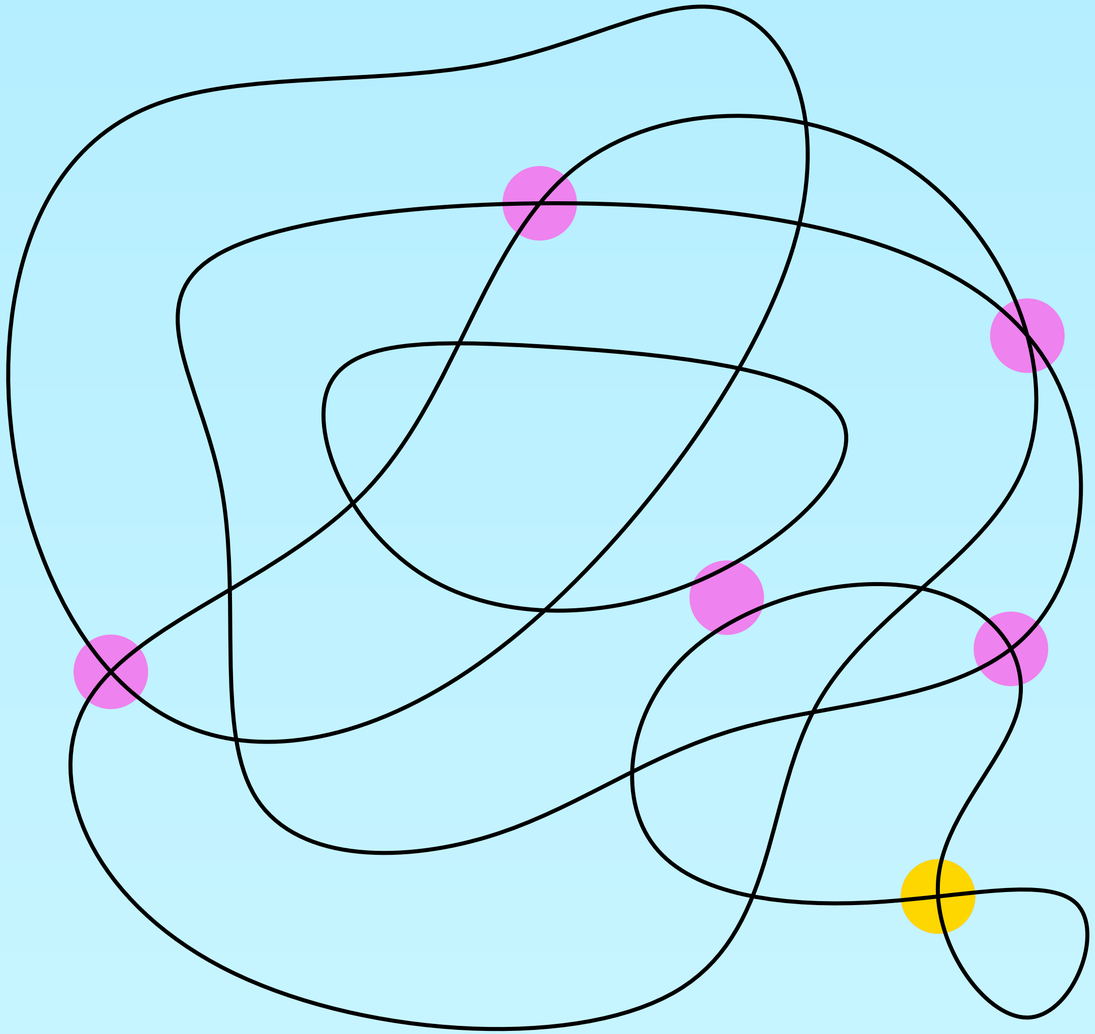
Fixed Parameter Tractability.

Perhaps the most natural parameter is the number of self-intersections or popular faces in the input.



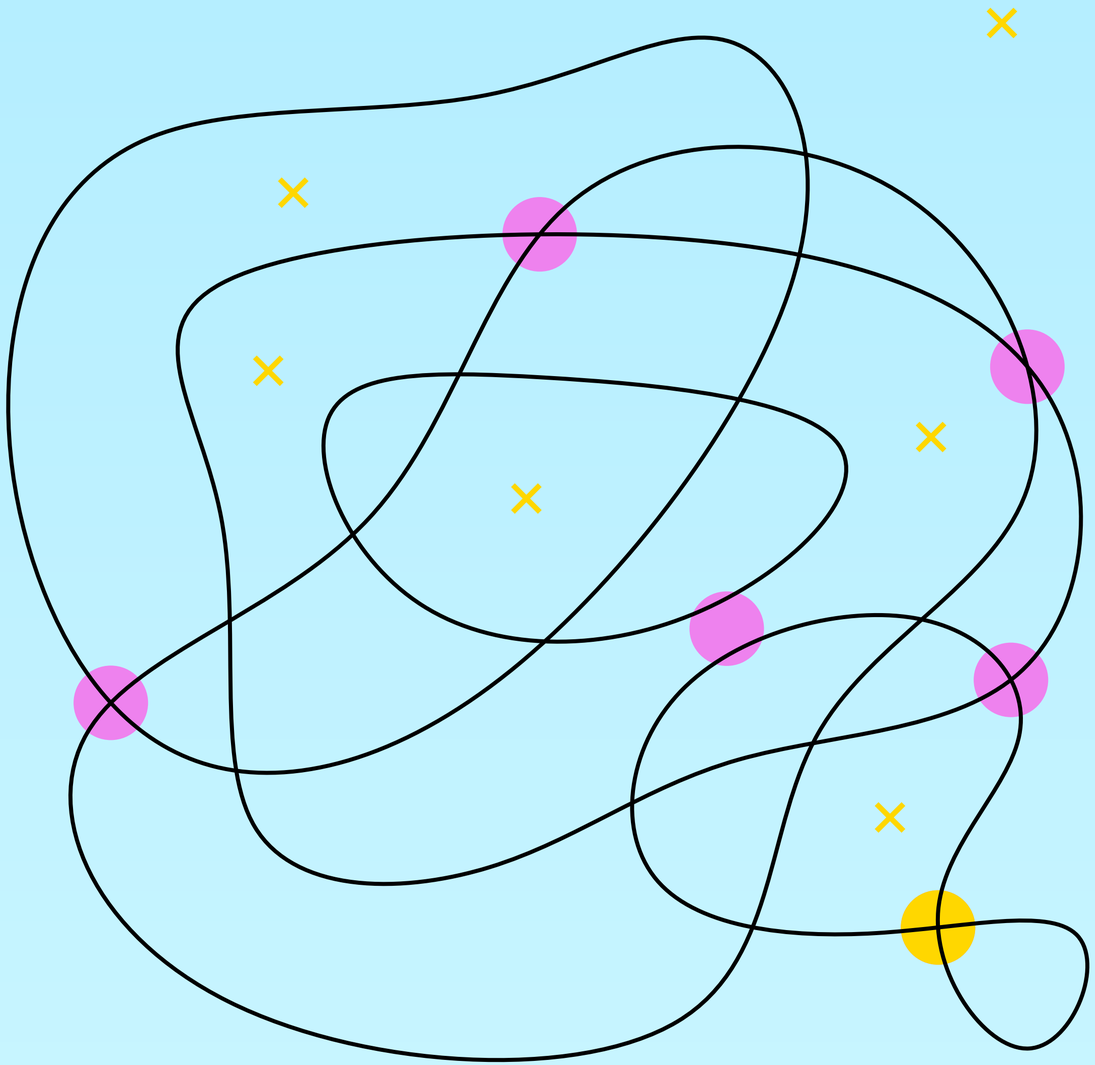
Fixed Parameter Tractability.

Perhaps the most natural parameter is the number of self-intersections or popular faces in the input.



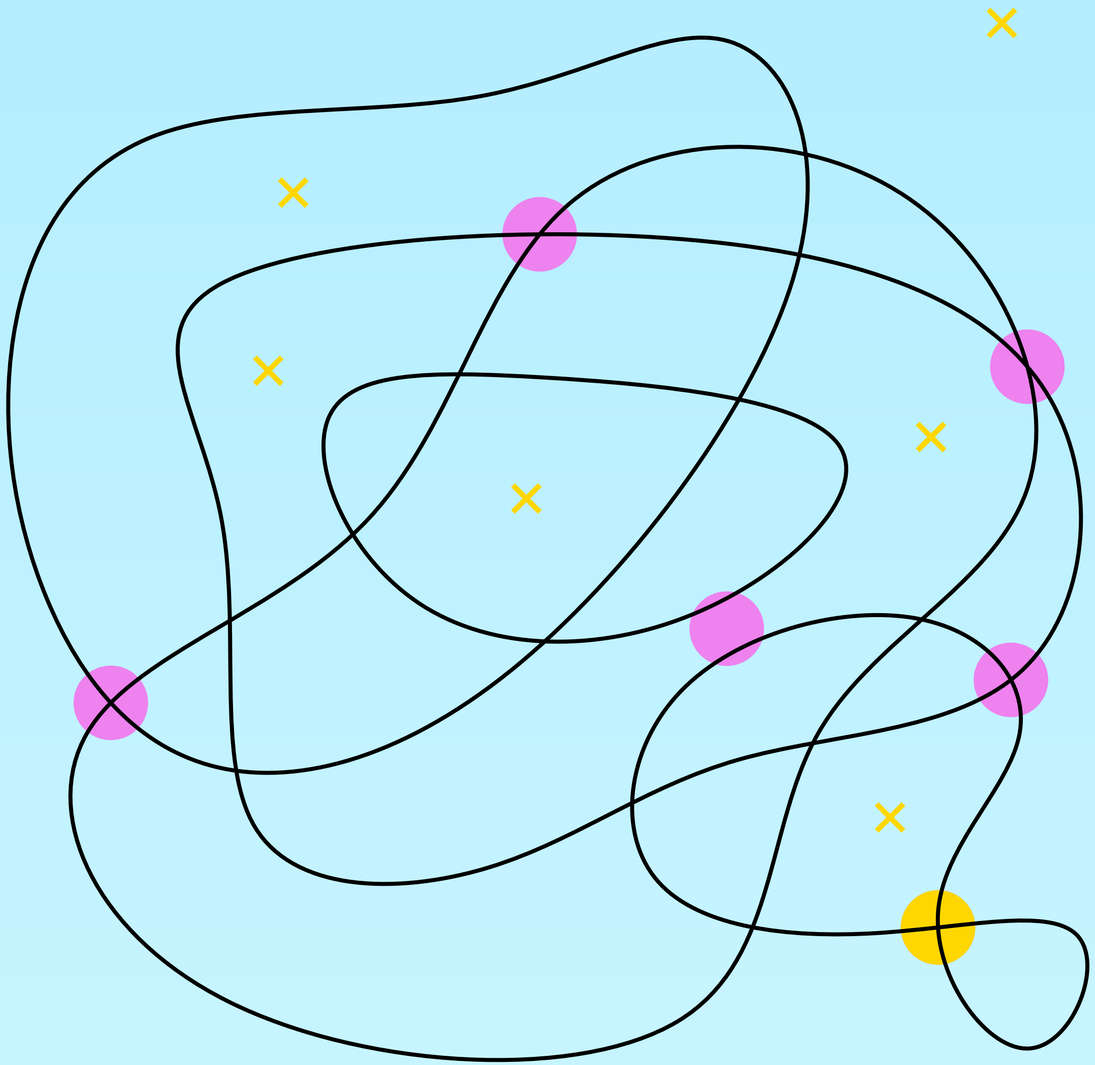
Fixed Parameter Tractability.

Perhaps the most natural parameter is the number of self-intersections or popular faces in the input.



Fixed Parameter Tractability.

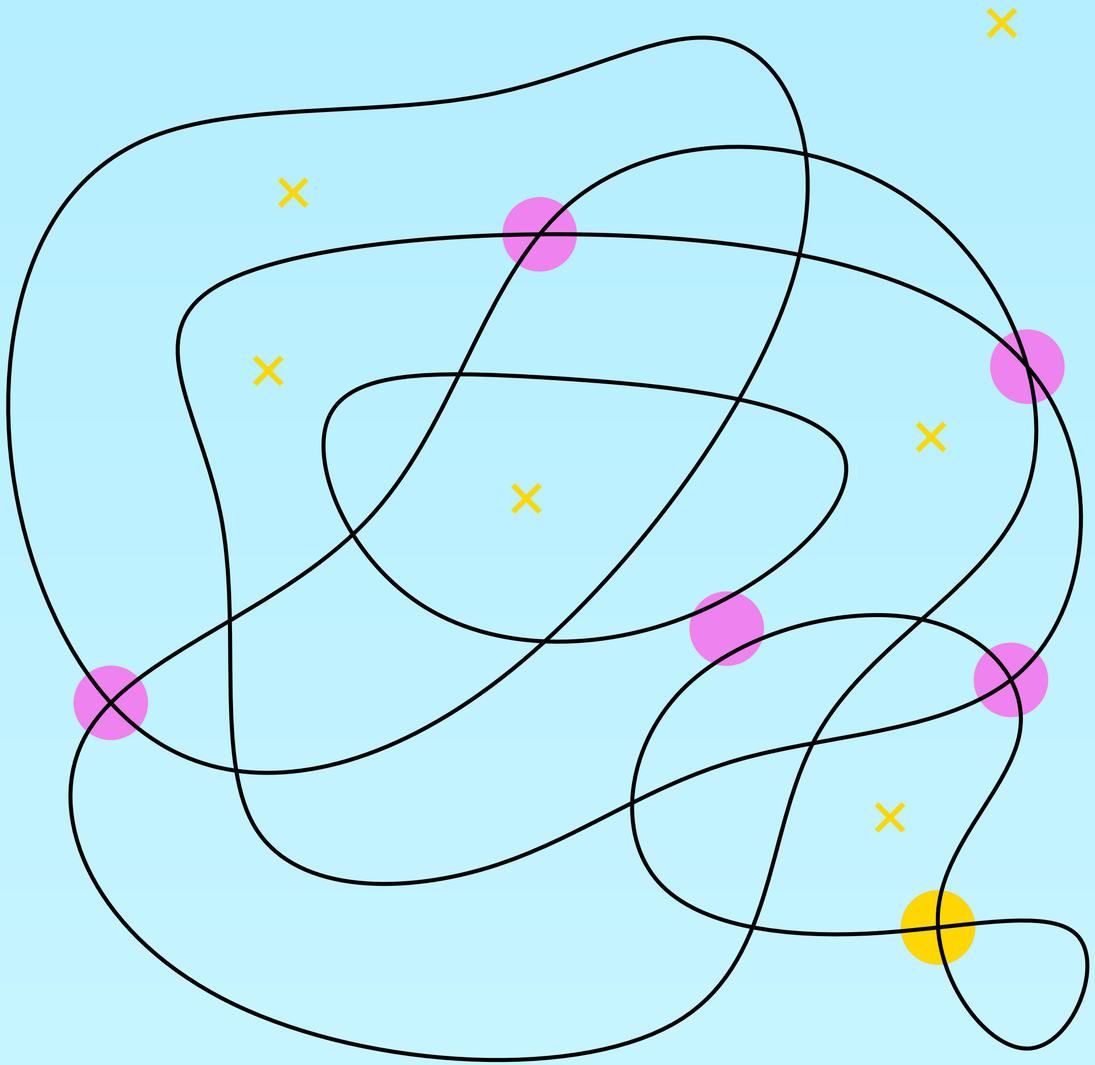
Perhaps the most natural parameter is the number of self-intersections or popular faces in the input.



Fixed Parameter Tractability.

Perhaps the most natural parameter is the number of self-intersections or popular faces in the input.

Unfortunately, our hardness constructions can easily be in initial state where this parameter is constant.

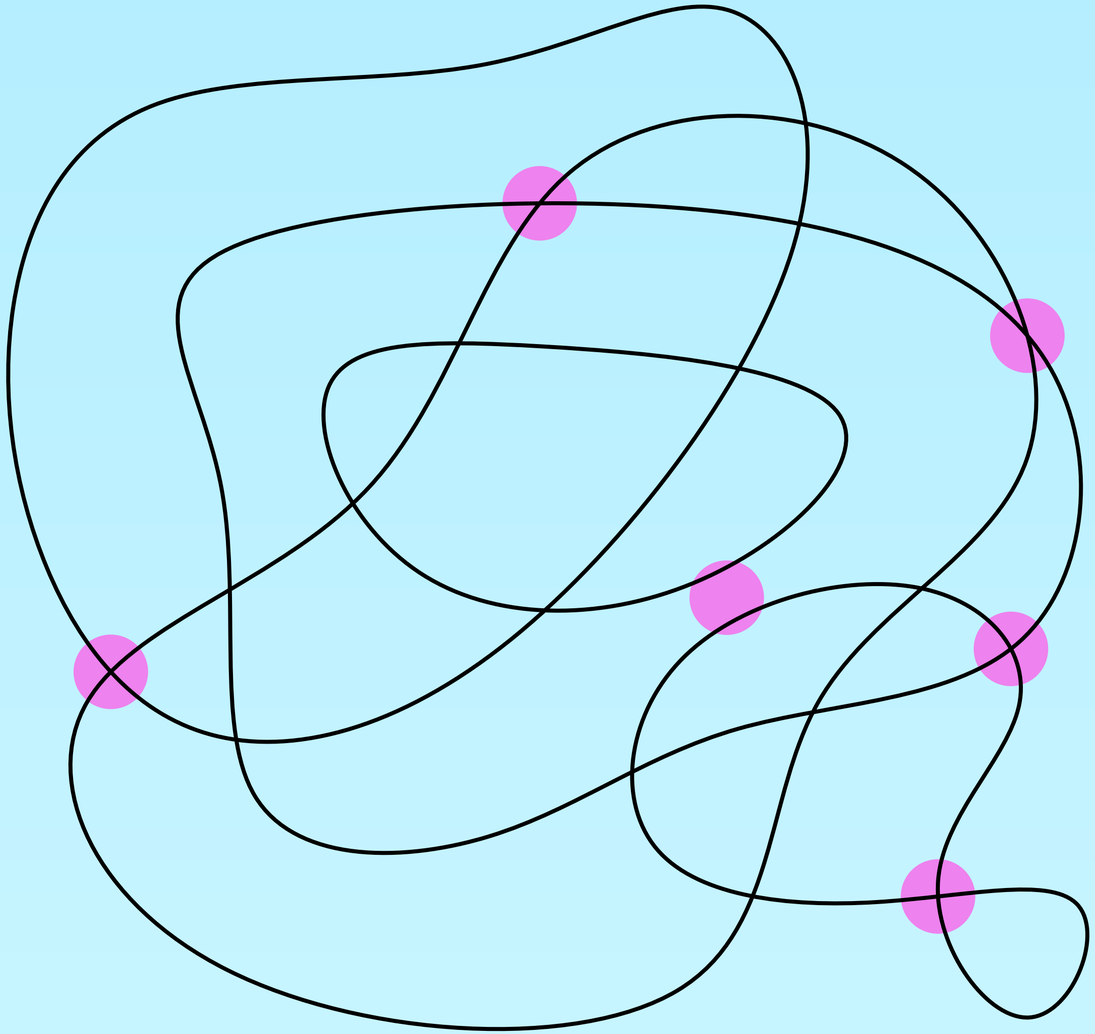


Fixed Parameter Tractability.

Perhaps the most natural parameter is the number of self-intersections or popular faces in the input.

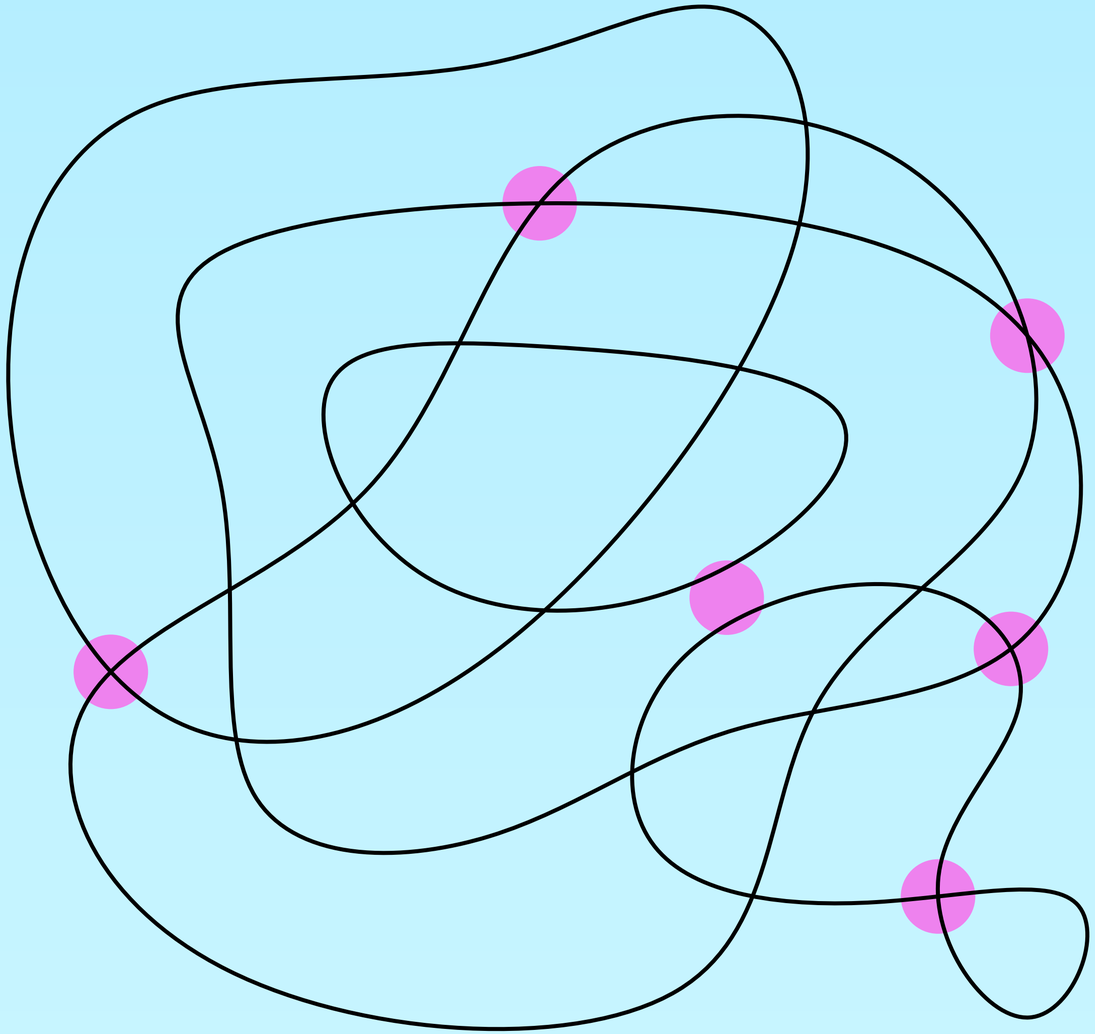
Unfortunately, our hardness constructions can easily be in initial state where this parameter is constant.

$\rightarrow$ not a good parameter.



Fixed Parameter Tractability.

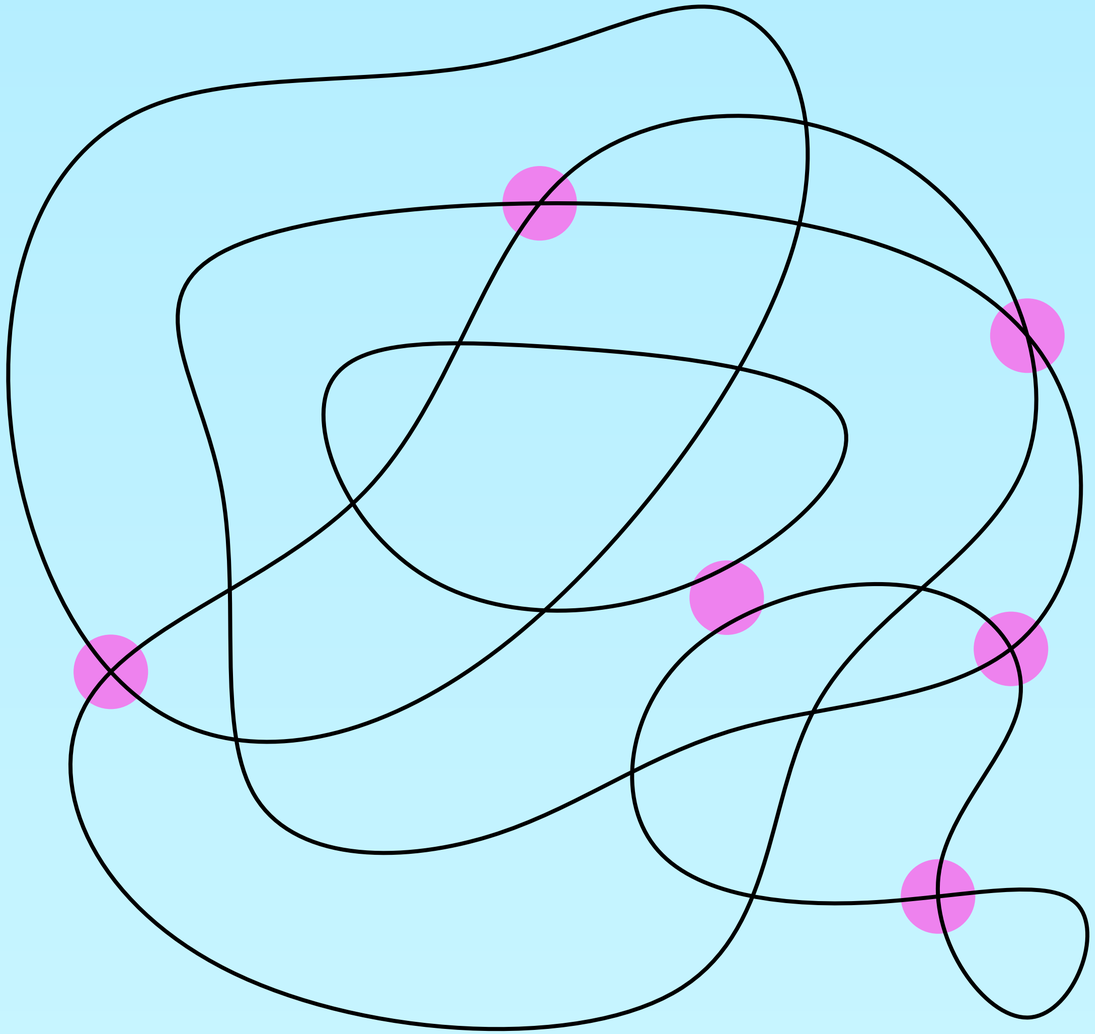
Another natural parameter is the number of switches.



Fixed Parameter Tractability.

Another natural parameter is the number of switches.

Only 3^k possible states $\rightarrow$ straightforward algorithm.

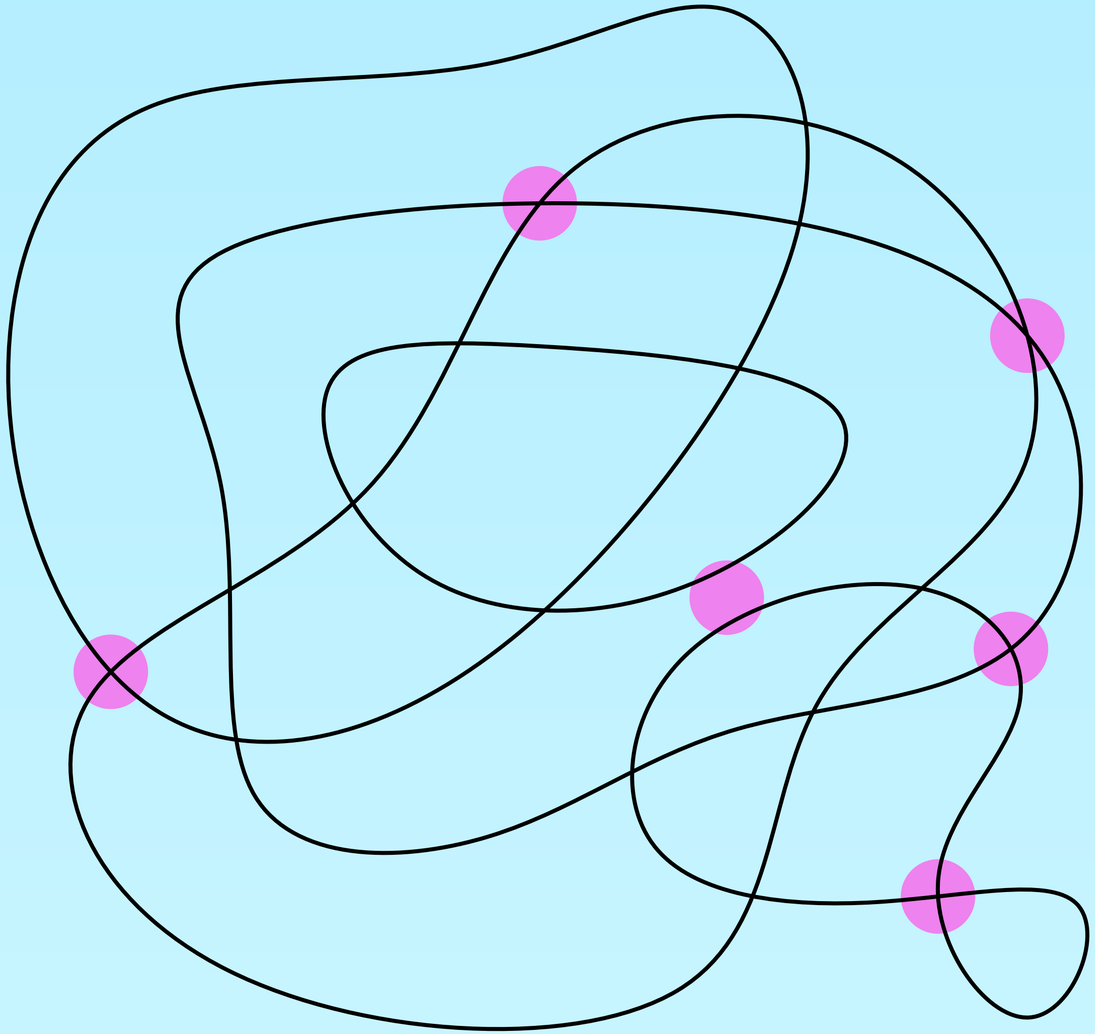


Fixed Parameter Tractability.

Another natural parameter is the number of switches.

Only 3^k possible states $\rightarrow$ straightforward algorithm.

However, ideally we want to have many switches for optimal flexibility.



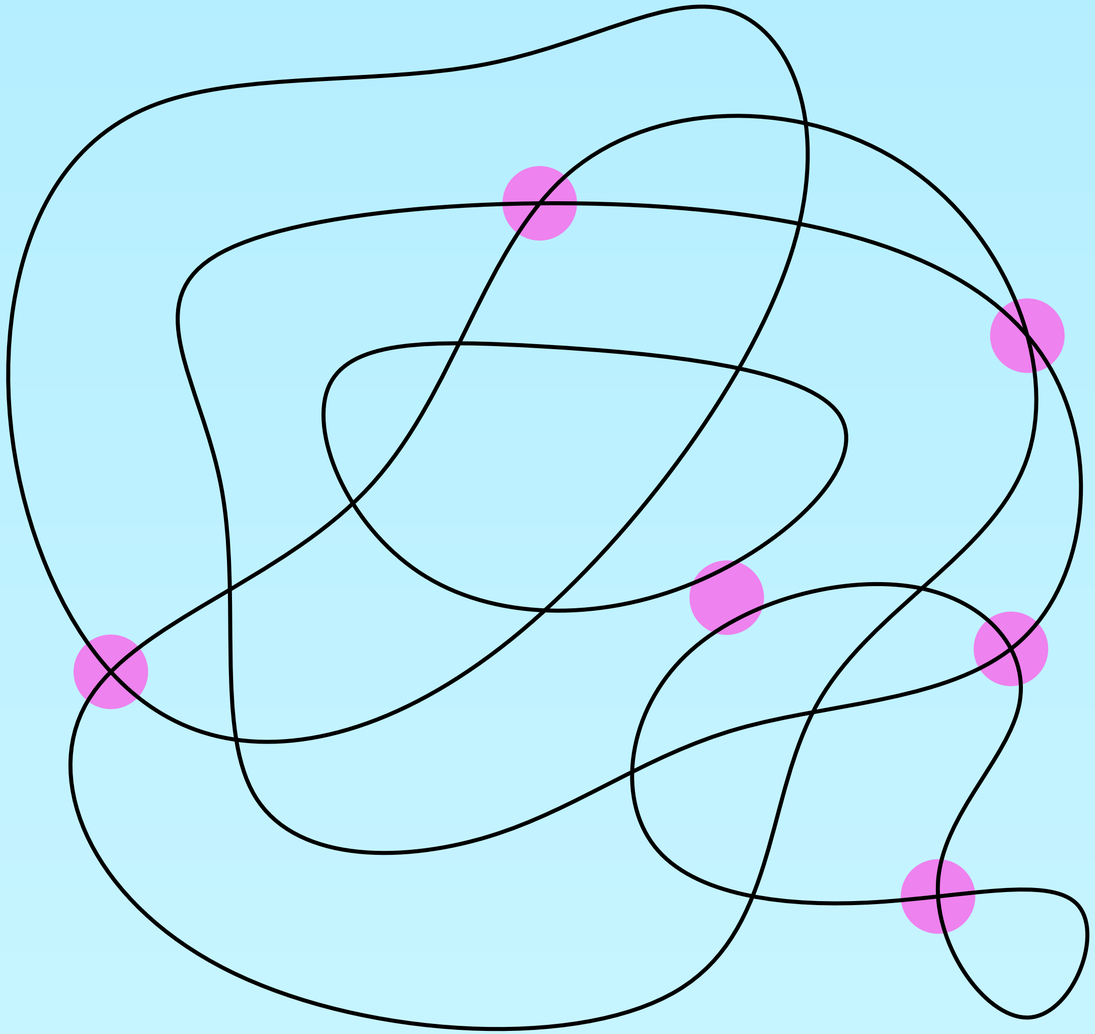
Fixed Parameter Tractability.

Another natural parameter is the number of switches.

Only 3^k possible states $\rightarrow$ straightforward algorithm.

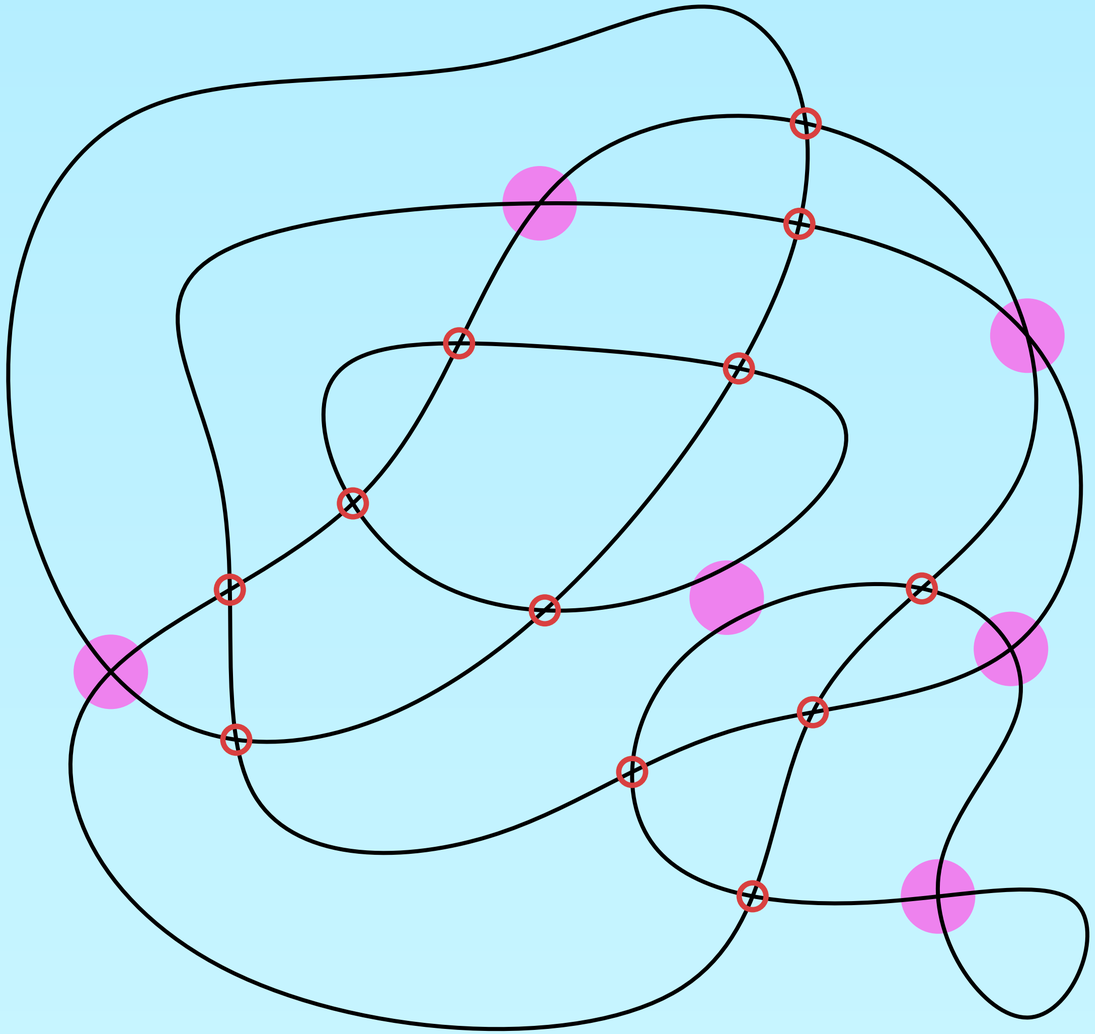
However, ideally we want to have many switches for optimal flexibility.

$\rightarrow$ also not a good parameter.



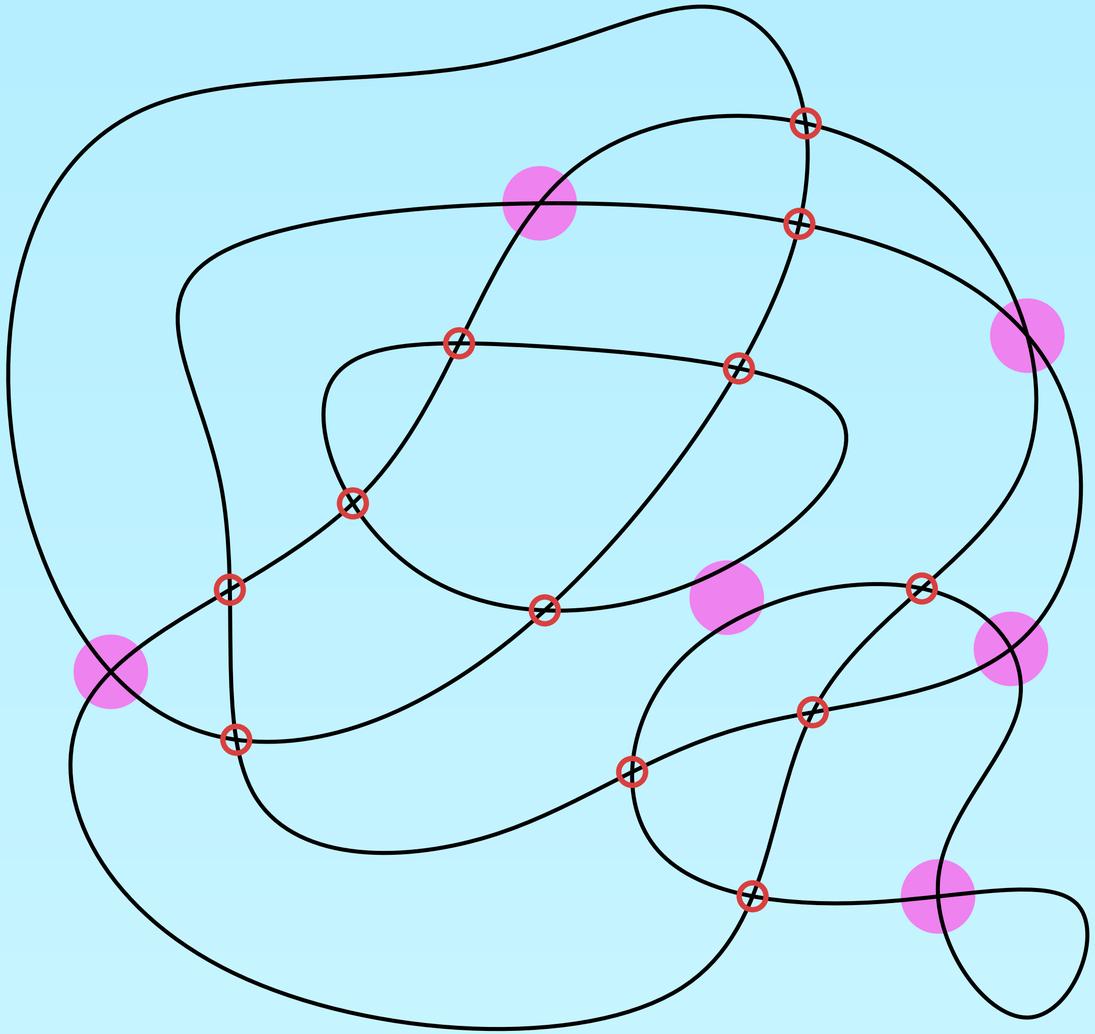
Fixed Parameter Tractability.

What we really want as a parameter is the number of non-switchable vertices.



Fixed Parameter Tractability.

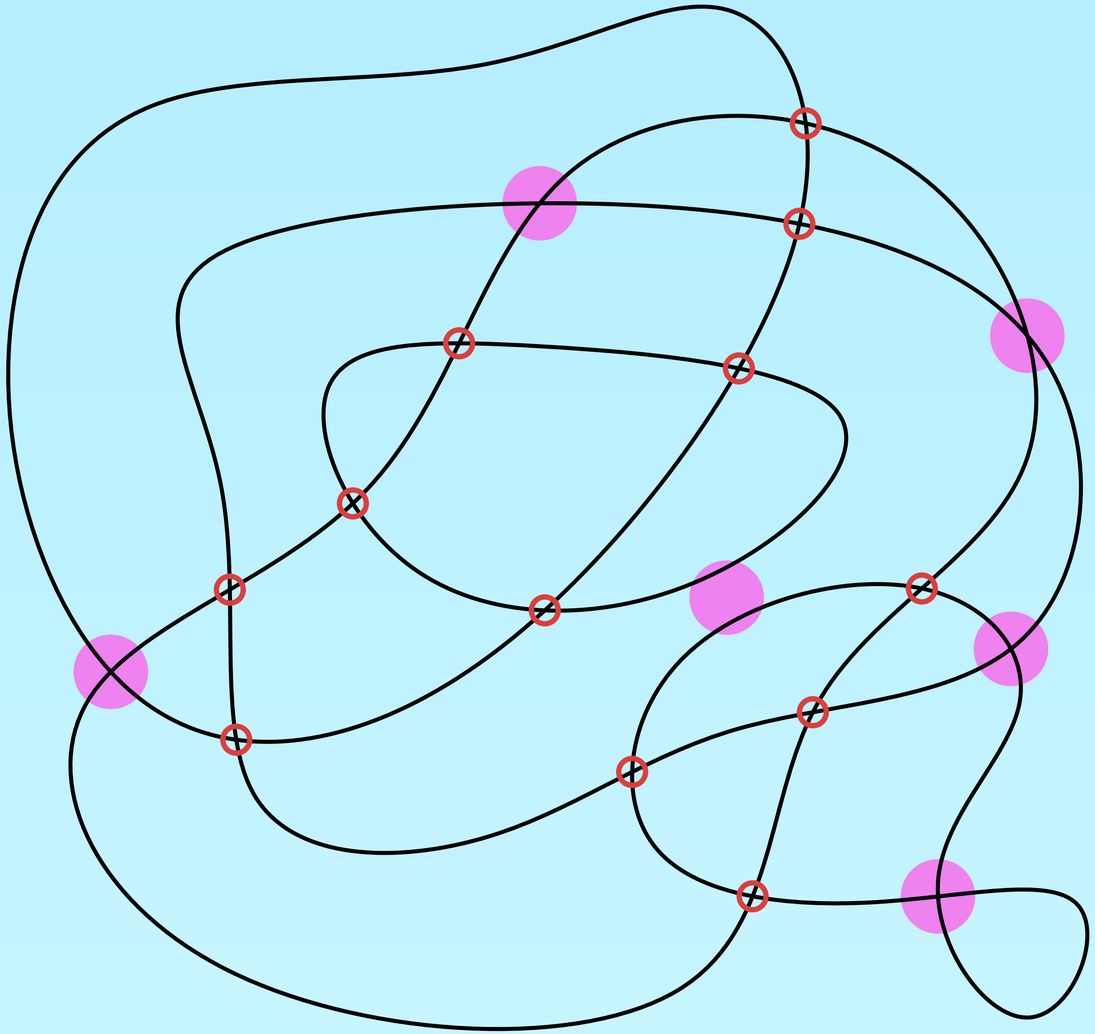
What we really want as a parameter is the number of non-switchable vertices.



Fixed Parameter Tractability.

What we really want as a parameter is the number of non-switchable vertices.

This parameter is actually interesting!



Fixed Parameter Tractability.

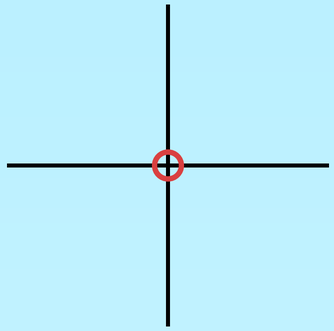
What we really want as a parameter is the number of non-switchable vertices.

This parameter is actually interesting!

Remember, when there are 0 non-switchable switches, the problem is trivial.

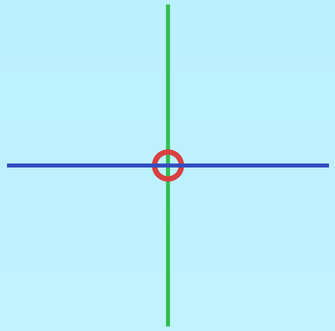
Fixed Parameter Tractability.

As a hint into what we can do: consider just one non-switchable vertex v.



Fixed Parameter Tractability.

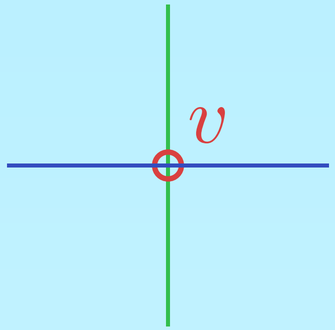
As a hint into what we can do: consider just one non-switchable vertex v.



Fixed Parameter Tractability.

As a hint into what we can do: consider just one non-switchable vertex v.

No self-intersections in the output means the two strands belong to different curves.

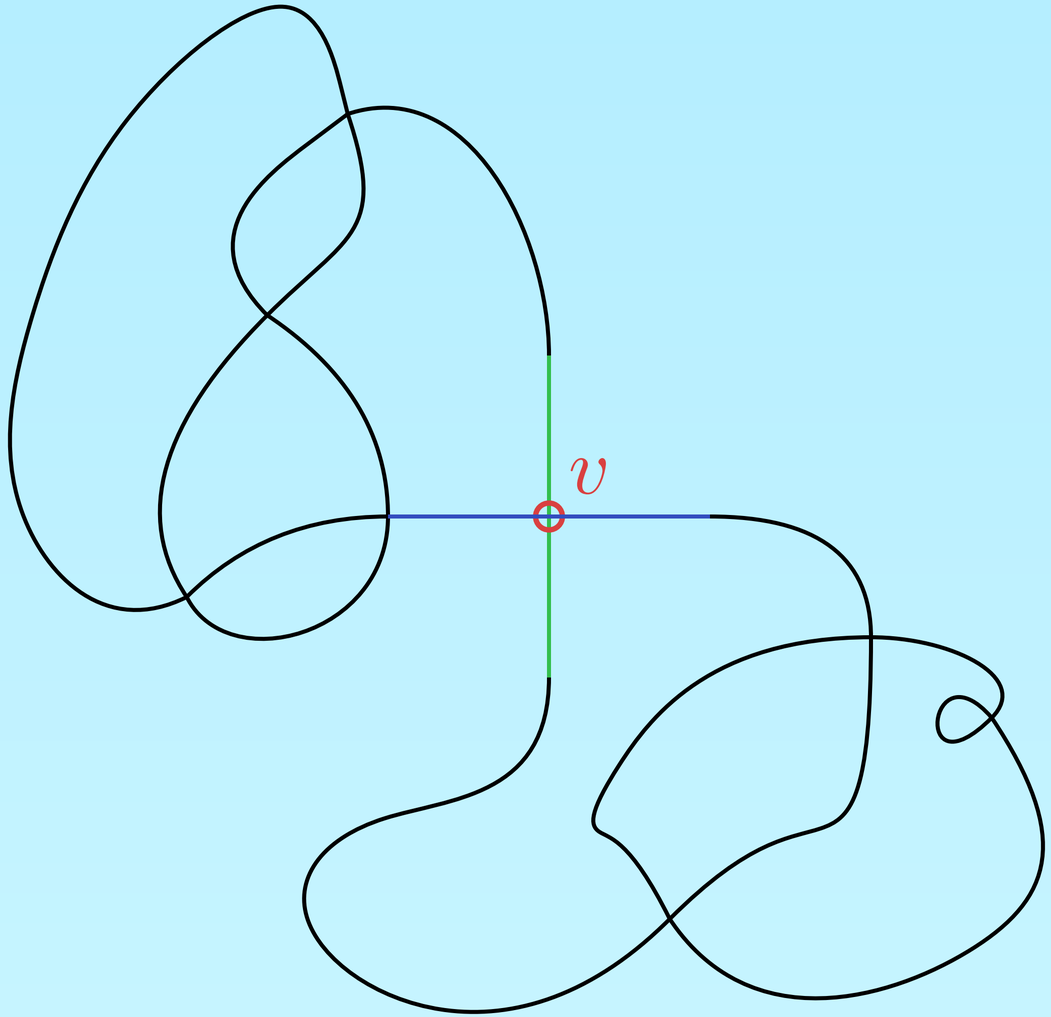


Fixed Parameter Tractability.

As a hint into what we can do: consider just one non-switchable vertex v.

No self-intersections in the output means the two strands belong to different curves.

Claim. This is possible if and only if the arrangement remains connected when we remove v.

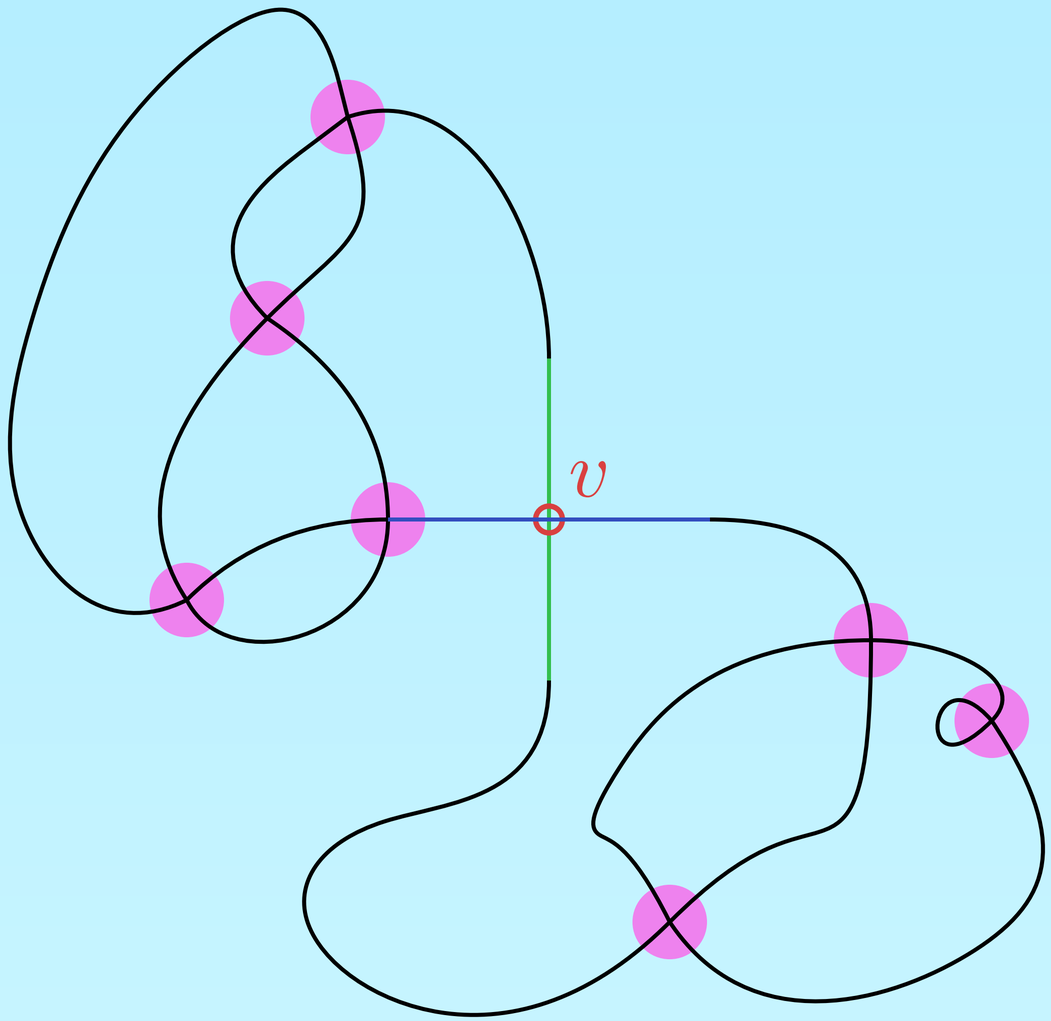


Fixed Parameter Tractability.

As a hint into what we can do: consider just one non-switchable vertex v.

No self-intersections in the output means the two strands belong to different curves.

Claim. This is possible if and only if the arrangement remains connected when we remove v.

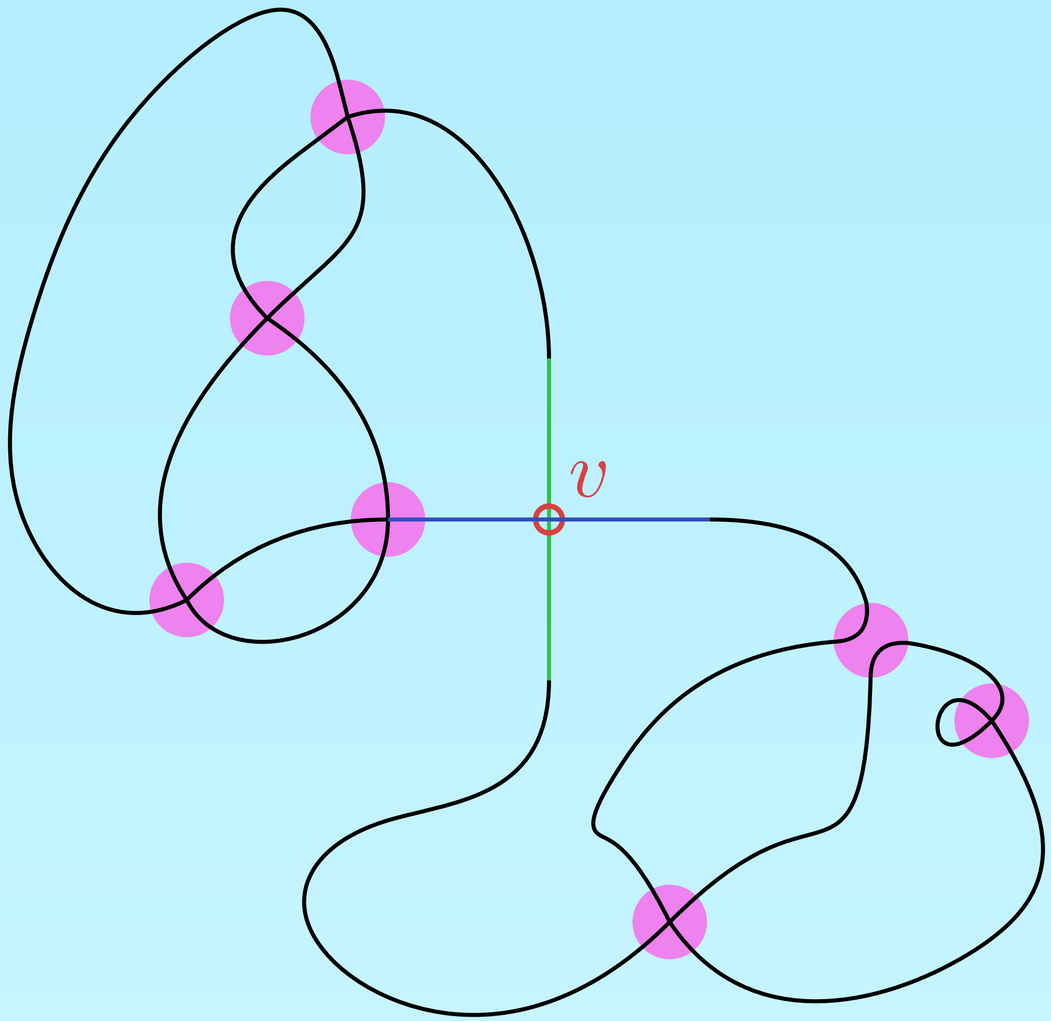


Fixed Parameter Tractability.

As a hint into what we can do: consider just one non-switchable vertex v.

No self-intersections in the output means the two strands belong to different curves.

Claim. This is possible if and only if the arrangement remains connected when we remove v.

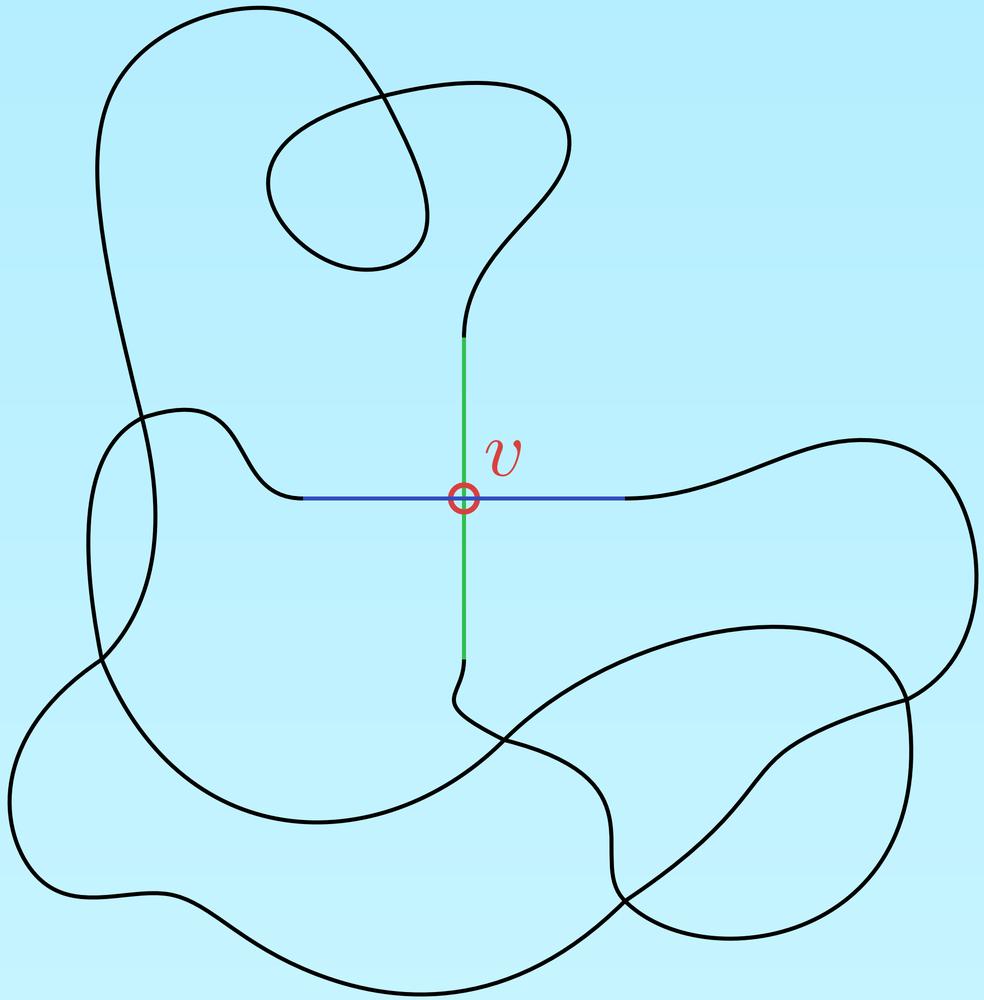


Fixed Parameter Tractability.

As a hint into what we can do: consider just one non-switchable vertex v.

No self-intersections in the output means the two strands belong to different curves.

Claim. This is possible if and only if the arrangement remains connected when we remove v.

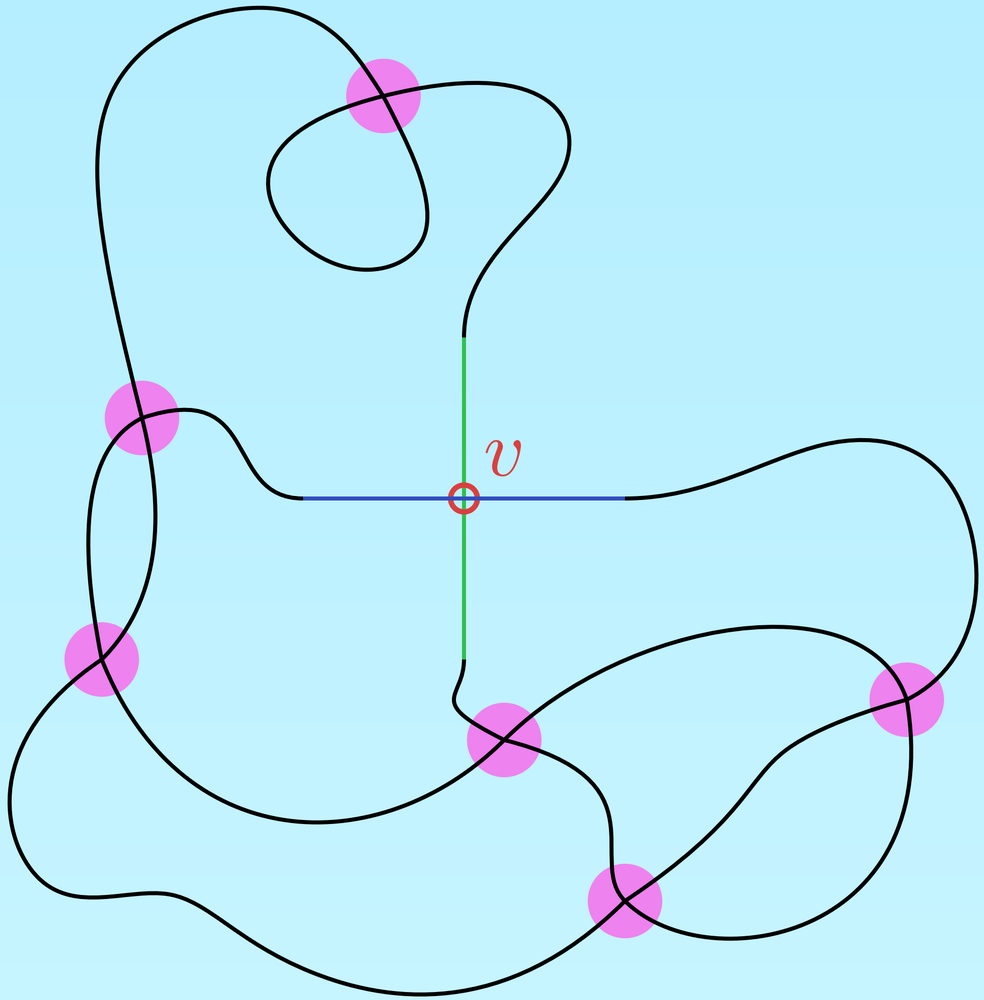


Fixed Parameter Tractability.

As a hint into what we can do: consider just one non-switchable vertex v.

No self-intersections in the output means the two strands belong to different curves.

Claim. This is possible if and only if the arrangement remains connected when we remove v.

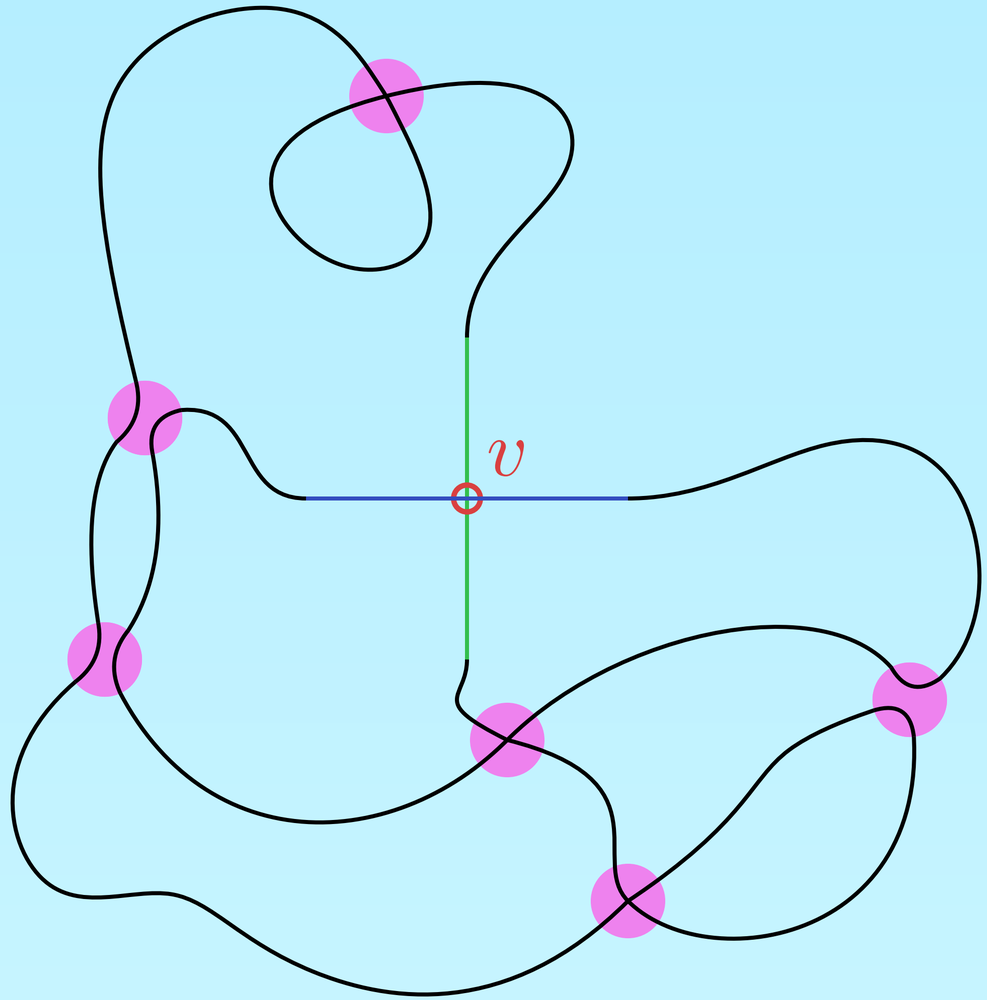


Fixed Parameter Tractability.

As a hint into what we can do: consider just one non-switchable vertex v.

No self-intersections in the output means the two strands belong to different curves.

Claim. This is possible if and only if the arrangement remains connected when we remove v.

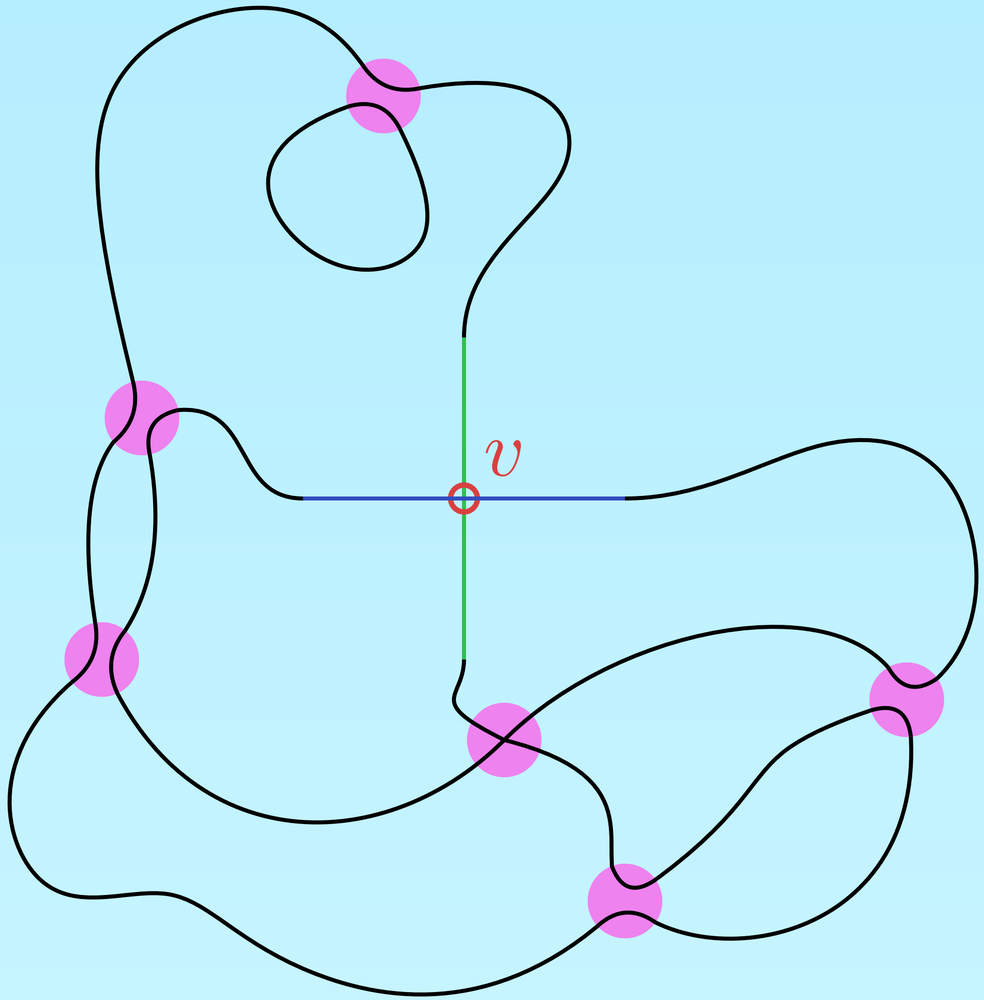


Fixed Parameter Tractability.

As a hint into what we can do: consider just one non-switchable vertex v.

No self-intersections in the output means the two strands belong to different curves.

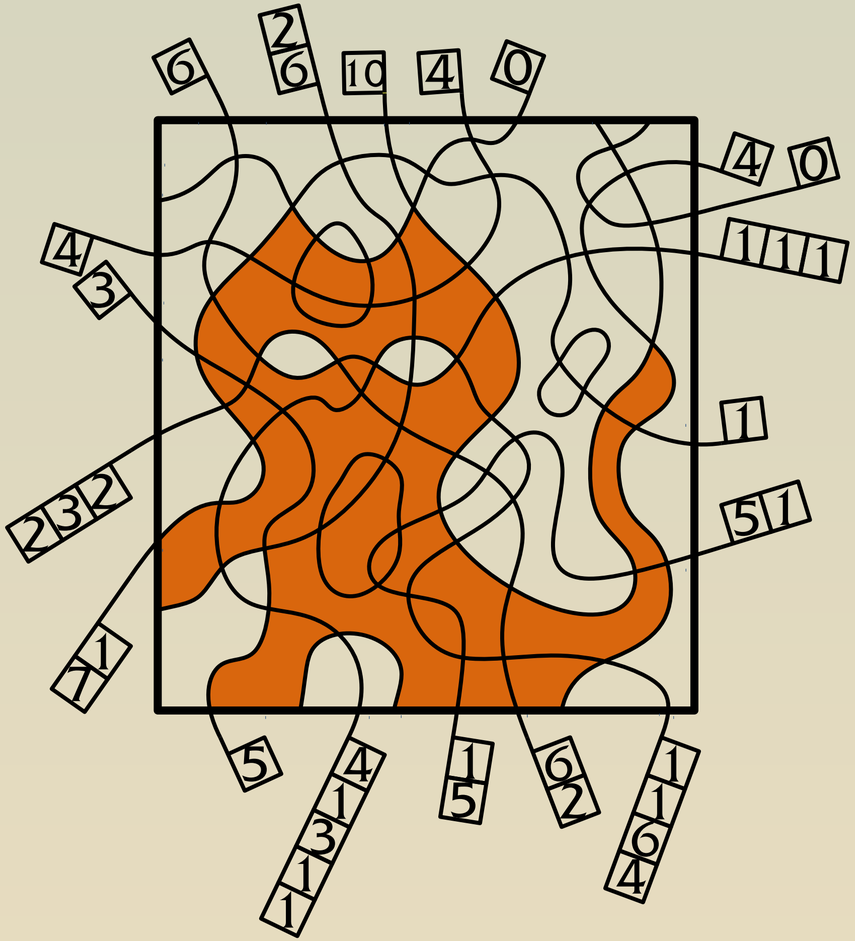
Claim. This is possible if and only if the arrangement remains connected when we remove v.



Fixed Parameter Tractability.

As a hint into what we can do: consider just one non-switchable vertex v.

No self-intersections in the output means the two strands belong to different curves.

Claim. This is possible if and only if the arrangement remains connected when we remove v.

Thank You!