Eliminating Crossings in Ordered Graphs

TCS Colloquium @ UJ — SWAT 2024

Akanksha Agrawal, Sergio Cabello, Michael Kaufmann,
Saket Saurabh, Roohani Sharma, Yushi Uno, **Alexander Wolff**

arxiv.org/abs/2404.09771

Graph Drawing: How to Deal with Crossings?

Many crossings typically make it hard to understand the drawing of a graph.

Graph Drawing: How to Deal with Crossings?

Many crossings typically make it hard to understand the drawing of a graph. Reduce number of crossings in drawings!

Graph Drawing: How to Deal with Crossings?

Many crossings typically make it hard to understand the drawing of a graph. Reduce number of crossings in drawings!

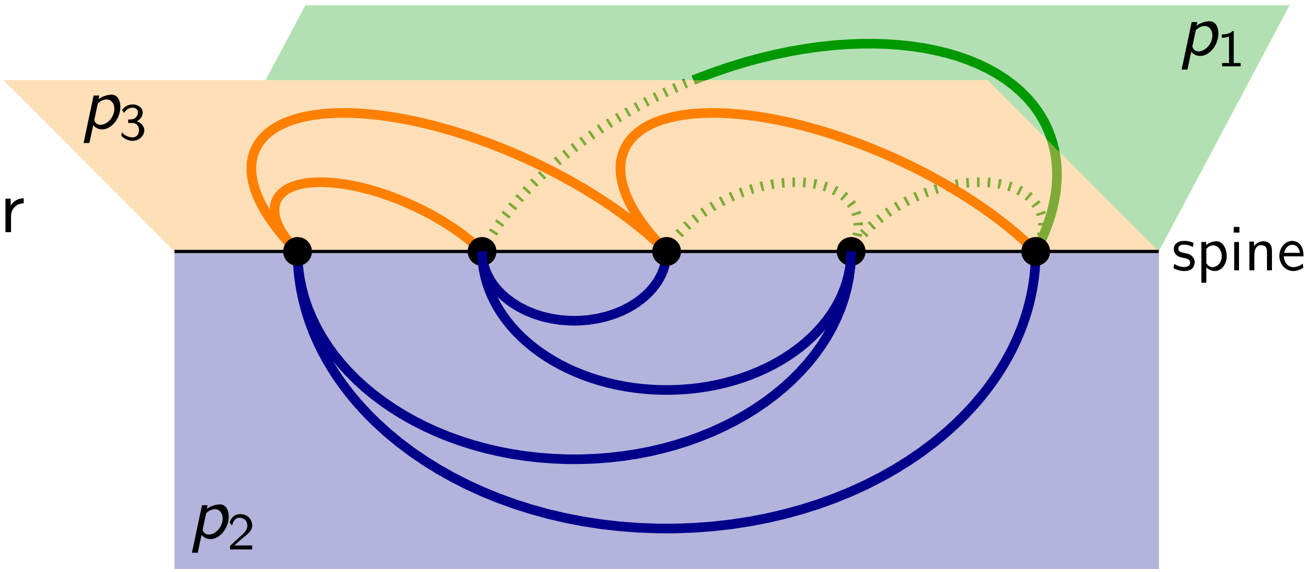
Another option: Remove smallest subset of vertices or edges s.t. the remaining graph can be drawn *without* crossings.

Graph Drawing: How to Deal with Crossings?

Many crossings typically make it hard to understand the drawing of a graph. Reduce number of crossings in drawings!

Another option: Remove smallest subset of vertices or edges s.t. the remaining graph can be drawn *without* crossings.

Our setting:
book embedding
with fixed vertex order

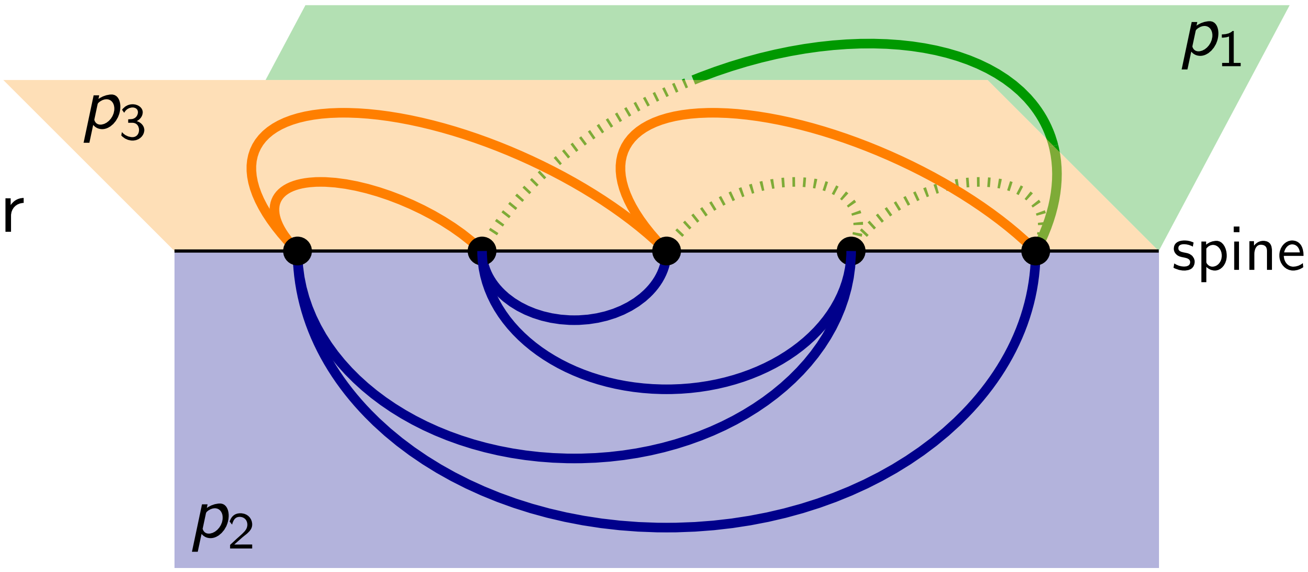


Graph Drawing: How to Deal with Crossings?

Many crossings typically make it hard to understand the drawing of a graph. Reduce number of crossings in drawings!

Another option: Remove smallest subset of vertices or edges s.t. the remaining graph can be drawn *without* crossings.

Our setting:
book embedding
with fixed vertex order



Our aim:
fast parametrized
exact algorithms

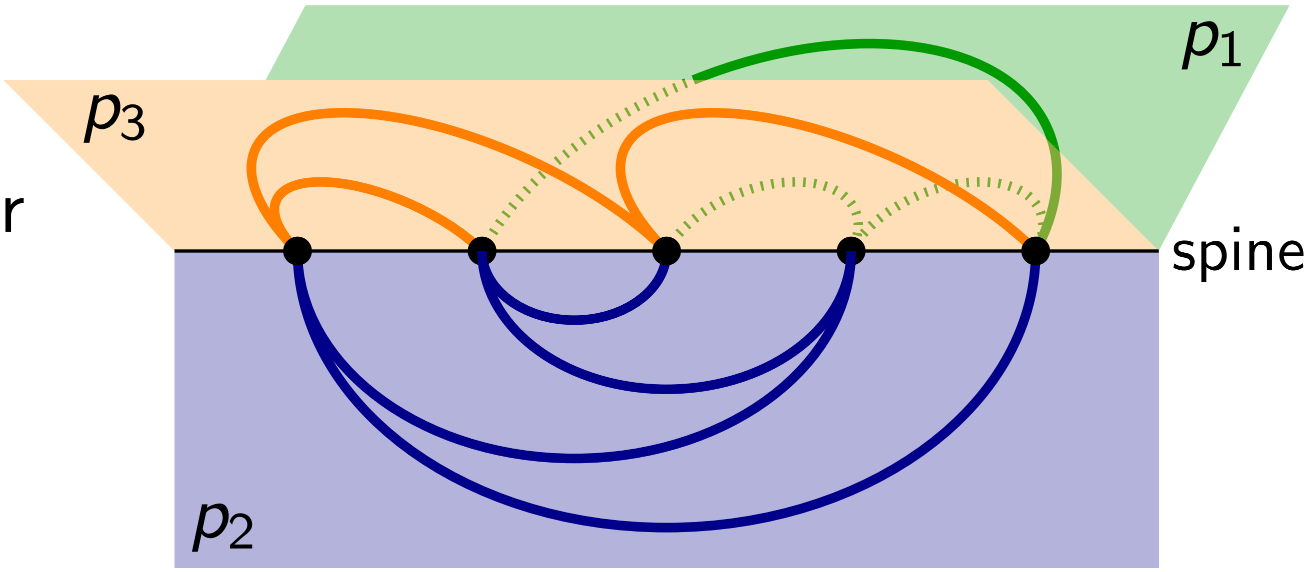
Graph Drawing: How to Deal with Crossings?

Many crossings typically make it hard to understand the drawing of a graph. Reduce number of crossings in drawings!

Another option: Remove smallest subset of vertices or edges s.t. the remaining graph can be drawn *without* crossings.

Our setting:
book embedding
with fixed vertex order

Our aim:
fast parametrized
exact algorithms



Yet another option: Remove part of every edge (e.g., middle half) → *partial edge drawings* (not today).

The Problem

EDGE DELETION TO p-PAGE d-PLANAR

Input:	ordered graph (G, σ), positive integers k, p, d.
Parameters:	k, p, d
Question:	Does there exist a set S of at most k edges of G such that $(G - S, \sigma)$ is p-page d-planar?

The Problem

EDGE DELETION TO p-PAGE d-PLANAR

Input:	ordered graph (G, σ), positive integers k, p, d.
Parameters:	k, p, d
Question:	Does there exist a set S of at most k edges of G such that $(G - S, \sigma)$ is p-page d-planar?

Disclaimer: We view p and d, though they appear in the problem name, not as constants, but as parameters.

The Problem

Edge Deletion to p-Page d-Planar

Input:	ordered graph (G, σ), positive integers k, p, d.
Parameters:	k, p, d
Question:	Does there exist a set S of at most k edges of G such that $(G - S, \sigma)$ is p-page d-planar?

Disclaimer: We view p and d, though they appear in the problem name, not as constants, but as parameters.

Examples: – What is the page number of K_5?

The Problem

Edge Deletion to p-Page d-Planar

Input: ordered graph (G, σ), positive integers k, p, d.
Parameters: k, p, d
Question: Does there exist a set S of at most k edges of G such that $(G - S, \sigma)$ is p-page d-planar?

Disclaimer: We view p and d, though they appear in the problem name, not as constants, but as parameters.

Examples: – What is the page number of K_5?

How many edges must we remove
– for a planar drawing of K_5 on 2 pages?

The Problem

EDGE DELETION TO p-PAGE d-PLANAR

Input: ordered graph (G, σ), positive integers k, p, d.
Parameters: k, p, d
Question: Does there exist a set S of at most k edges of G such that $(G - S, \sigma)$ is p-page d-planar?

Disclaimer: We view p and d, though they appear in the problem name, not as constants, but as parameters.

Examples: – What is the page number of K_5?

How many edges must we remove
– for a planar drawing of K_5 on 2 pages?
– for a 1-planar drawing of K_5 on 2 pages?

The Problem

Edge Deletion to p-Page d-Planar

Input: ordered graph (G, σ), positive integers k, p, d.
Parameters: k, p, d
Question: Does there exist a set S of at most k edges of G such that $(G - S, \sigma)$ is p-page d-planar?

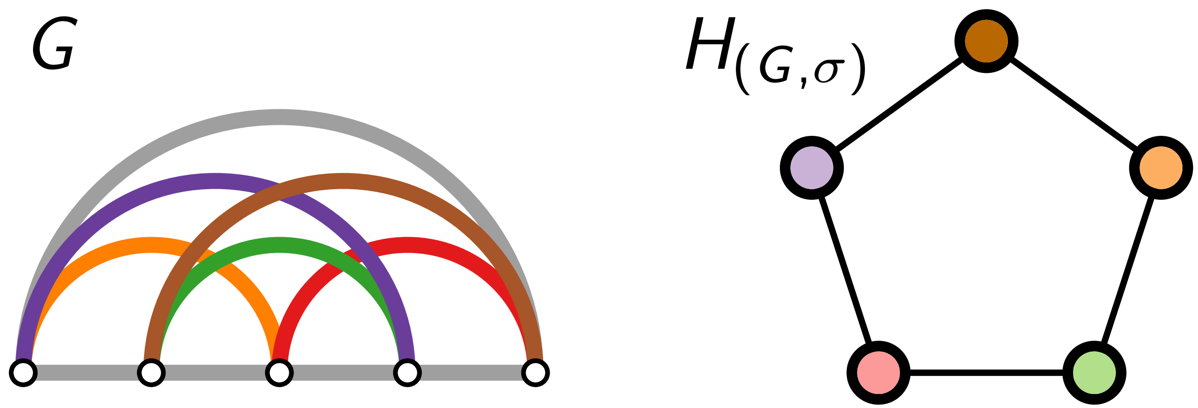
Disclaimer: We view p and d, though they appear in the problem name, not as constants, but as parameters.

Examples: – What is the page number of K_5?

How many edges must we remove
– for a planar drawing of K_5 on 2 pages?
– for a 1-planar drawing of K_5 on 2 pages?
– for a 2-planar drawing of K_5 on 1 page?

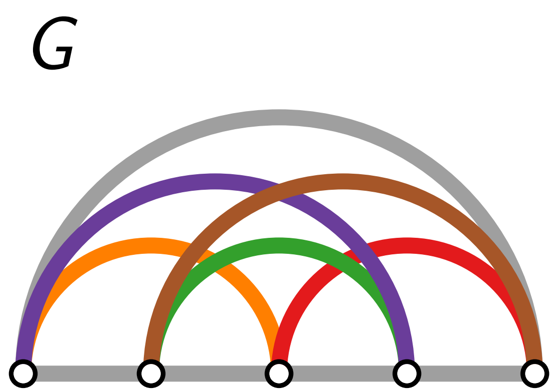
Another Way to See Things: Conflict Graph

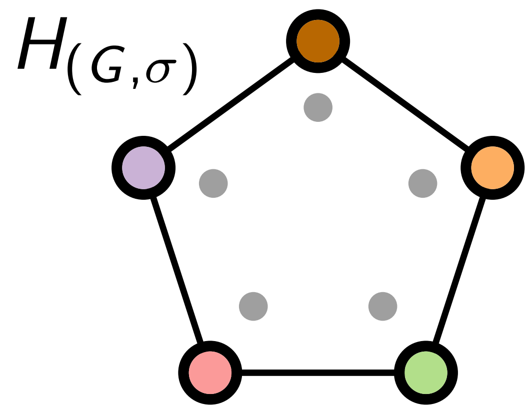
Given an ordered graph (G, σ), its *conflict graph* $H_{(G,\sigma)}$ is the graph that has a vertex for each edge of G and an edge for each pair of crossing edges of G.



Another Way to See Things: Conflict Graph

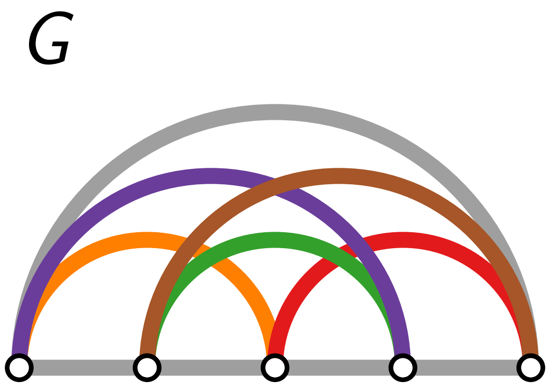
Given an ordered graph (G, σ), its *conflict graph* $H_{(G,\sigma)}$ is the graph that has a vertex for each edge of G and an edge for each pair of crossing edges of G.

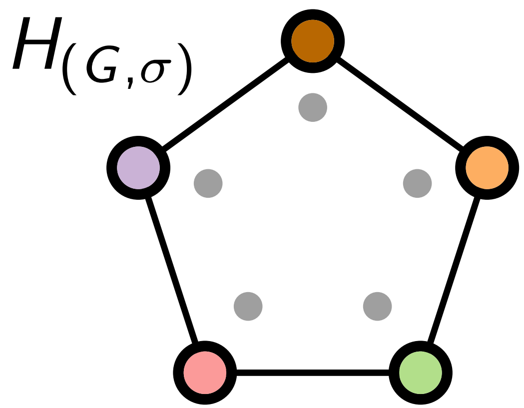




Another Way to See Things: Conflict Graph

Given an ordered graph (G, σ), its *conflict graph* $H_{(G,\sigma)}$ is the graph that has a vertex for each edge of G and an edge for each pair of crossing edges of G.

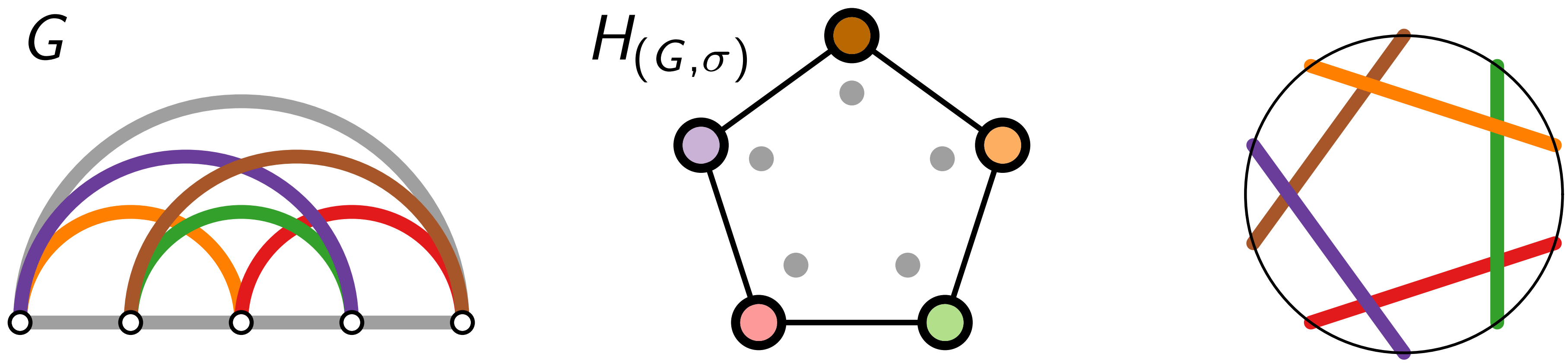




$H_{(G,\sigma)}$ is a *circle graph*, that is, the intersection graph of chords of a circle.

Another Way to See Things: Conflict Graph

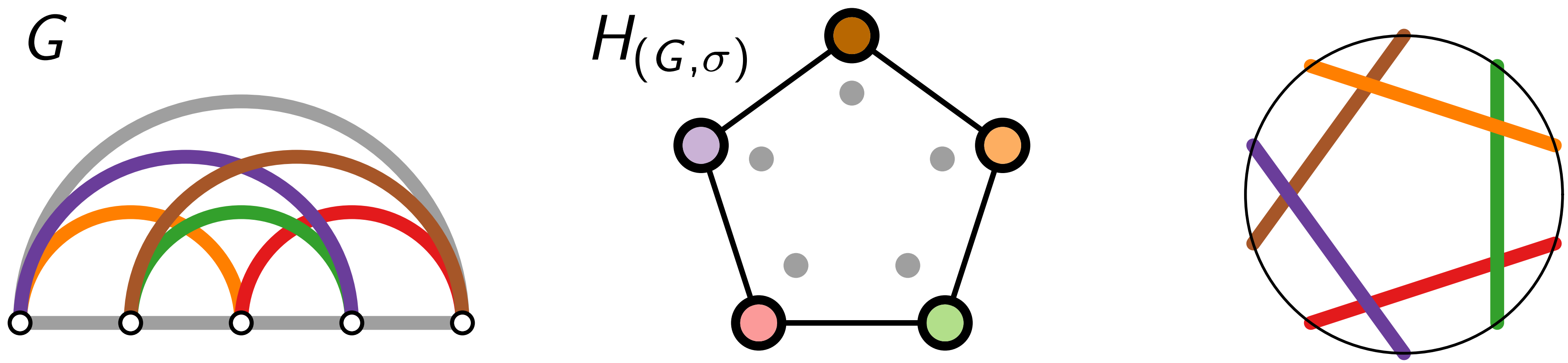
Given an ordered graph (G, σ), its *conflict graph* $H_{(G,\sigma)}$ is the graph that has a vertex for each edge of G and an edge for each pair of crossing edges of G.



$H_{(G,\sigma)}$ is a *circle graph*, that is, the intersection graph of chords of a circle.

Another Way to See Things: Conflict Graph

Given an ordered graph (G, σ), its *conflict graph* $H_{(G,\sigma)}$ is the graph that has a vertex for each edge of G and an edge for each pair of crossing edges of G.

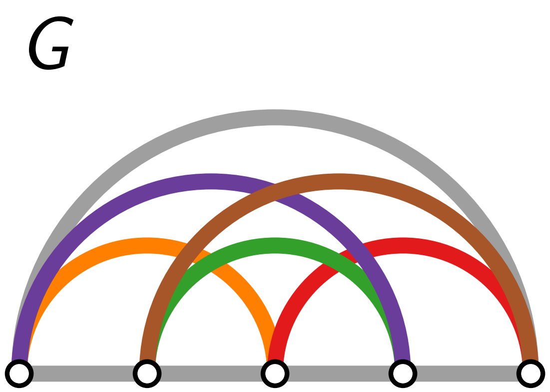


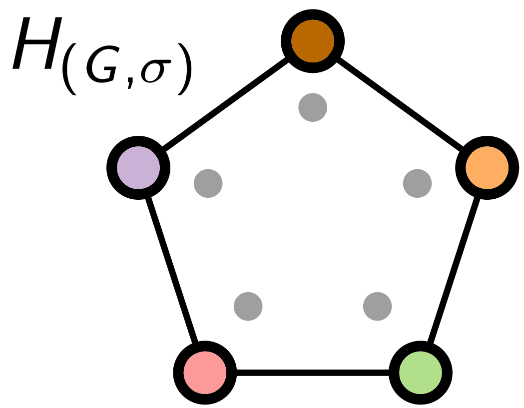
$H_{(G,\sigma)}$ is a *circle graph*, that is, the intersection graph of chords of a circle.

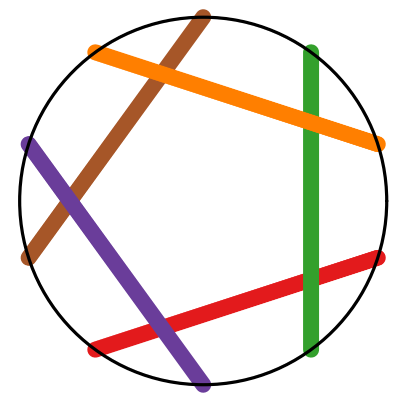
So Edge Deletion to **1**-Page d-Planar is the same as Vertex Deletion to Degree-d (in circle graphs).

Another Way to See Things: Conflict Graph

Given an ordered graph (G, σ), its *conflict graph* $H_{(G,\sigma)}$ is the graph that has a vertex for each edge of G and an edge for each pair of crossing edges of G.







$H_{(G,\sigma)}$ is a *circle graph*, that is, the intersection graph of chords of a circle.

So Edge Deletion to **1**-Page d-Planar is the same as Vertex Deletion to Degree-d (in circle graphs).

For general graphs, this admits a quadratic kernel. [Xiao, 2017]

Related Work (I)

Testing whether (G, σ) has (fixed-vertex-order) page number p (without edge deletions) is equivalent to

Related Work (I)

Testing whether (G, σ) has (fixed-vertex-order) page number p (without edge deletions) is equivalent to p-colorability of $H_{(G,\sigma)}$.

Related Work (I)

Testing whether (G, σ) has (fixed-vertex-order) page number p (without edge deletions) is equivalent to p-colorability of $H_{(G,\sigma)}$.

So for $p = 2$, it suffices to test whether $H_{(G,\sigma)}$ is bipartite.

Related Work (I)

Testing whether (G, σ) has (fixed-vertex-order) page number p (without edge deletions) is equivalent to p-colorability of $H_{(G,\sigma)}$.

So for $p = 2$, it suffices to test whether $H_{(G,\sigma)}$ is bipartite.

For $p = 4$, Unger [1988] showed that the problem is NP-hard.

Related Work (I)

Testing whether (G, σ) has (fixed-vertex-order) page number p (without edge deletions) is equivalent to p-colorability of $H_{(G,\sigma)}$.

So for $p = 2$, it suffices to test whether $H_{(G,\sigma)}$ is bipartite.

For $p = 4$, Unger [1988] showed that the problem is NP-hard.

For $p = 3$, he [1992] claimed an efficient solution, but his approach is incomplete. [Bachmann et al., GD 2023]

Related Work (I)

Testing whether (G, σ) has (fixed-vertex-order) page number p (without edge deletions) is equivalent to p-colorability of $H_{(G,\sigma)}$.

So for $p = 2$, it suffices to test whether $H_{(G,\sigma)}$ is bipartite.

For $p = 4$, Unger [1988] showed that the problem is NP-hard.

For $p = 3$, he [1992] claimed an efficient solution, but his approach is incomplete. [Bachmann et al., GD 2023]

Edge Deletion to p-Page Planar: special case $d = 0$.

Related Work (I)

Testing whether (G, σ) has (fixed-vertex-order) page number p (without edge deletions) is equivalent to p-colorability of $H_{(G,\sigma)}$.

So for $p = 2$, it suffices to test whether $H_{(G,\sigma)}$ is bipartite.

For $p = 4$, Unger [1988] showed that the problem is NP-hard.

For $p = 3$, he [1992] claimed an efficient solution, but his approach is incomplete. [Bachmann et al., GD 2023]

Edge Deletion to p-Page Planar: special case $d = 0$.
This is in $H_{(G,\sigma)}$.

Related Work (I)

Testing whether (G, σ) has (fixed-vertex-order) page number p (without edge deletions) is equivalent to p-colorability of $H_{(G,\sigma)}$.

So for $p = 2$, it suffices to test whether $H_{(G,\sigma)}$ is bipartite.

For $p = 4$, Unger [1988] showed that the problem is NP-hard.

For $p = 3$, he [1992] claimed an efficient solution, but his approach is incomplete. [Bachmann et al., GD 2023]

Edge Deletion to p-Page Planar: special case $d = 0$.
This is Vertex Deletion to p-Colorability in $H_{(G,\sigma)}$.

Related Work (I)

Testing whether (G, σ) has (fixed-vertex-order) page number p (without edge deletions) is equivalent to p-colorability of $H_{(G,\sigma)}$.

So for $p = 2$, it suffices to test whether $H_{(G,\sigma)}$ is bipartite.

For $p = 4$, Unger [1988] showed that the problem is NP-hard.

For $p = 3$, he [1992] claimed an efficient solution, but his approach is incomplete. [Bachmann et al., GD 2023]

EDGE DELETION TO p-PAGE PLANAR: special case $d = 0$.
This is VERTEX DELETION TO p-COLORABILITY in $H_{(G,\sigma)}$.

$p = 1$: MIS in circle graphs – quadratic time. [Valiante 2003]

Related Work (I)

Testing whether (G, σ) has (fixed-vertex-order) page number p (without edge deletions) is equivalent to p-colorability of $H_{(G,\sigma)}$.

So for $p = 2$, it suffices to test whether $H_{(G,\sigma)}$ is bipartite.

For $p = 4$, Unger [1988] showed that the problem is NP-hard.

For $p = 3$, he [1992] claimed an efficient solution, but his approach is incomplete. [Bachmann et al., GD 2023]

EDGE DELETION TO p-PAGE PLANAR: special case $d = 0$.
This is VERTEX DELETION TO p-COLORABILITY in $H_{(G,\sigma)}$.

$p = 1$: MIS in circle graphs – quadratic time. [Valiante 2003]

$p = 2$: ODD CYCLE TRANSVERSAL in circle graphs – FPT [Reed, Smith, Vetta, 2004]

Related Work (II)

Fixed-Order Page Number

Related Work (II)

FIXED-ORDER PAGE NUMBER can be solved in $2^{O(\mathrm{vc}^3)}n$ time

[Bhore, Ganian, Montecchiani, Nöllenburg, 2020]

Related Work (II)

Fixed-Order Page Number can be solved in $2^{O(\mathrm{vc}^3)} n$ time [Bhore, Ganian, Montecchiani, Nöllenburg, 2020]

and in $2^{O(\mathrm{pw}^2)} n$ time, where pw is the pathwidth of the *ordered* graph, which is not bounded by vc. [Liu, Chen, Huang, Wang, 2021]

Related Work (II)

FIXED-ORDER PAGE NUMBER can be solved in $2^{O(\mathrm{vc}^3)} n$ time

[Bhore, Ganian, Montecchiani, Nöllenburg, 2020]

and in $2^{O(\mathrm{pw}^2)} n$ time, where pw is the pathwidth of the *ordered* graph, which is not bounded by vc.

[Liu, Chen, Huang, Wang, 2021]

FIXED-ORDER BOOK DRAWING – testing if there is a p-page d-planar drawing of (G, σ)

Related Work (II)

FIXED-ORDER PAGE NUMBER can be solved in $2^{O(\mathrm{vc}^3)} n$ time [Bhore, Ganian, Montecchiani, Nöllenburg, 2020] and in $2^{O(\mathrm{pw}^2)} n$ time, where pw is the pathwidth of the *ordered* graph, which is not bounded by vc. [Liu, Chen, Huang, Wang, 2021]

FIXED-ORDER BOOK DRAWING – testing if there is a p-page d-planar drawing of (G, σ) – can be solved in $(d+2)^{O(\mathrm{vc}^3)} n$ or in $(d+2)^{O(\mathrm{pw}^2)} n$ time. [Liu, Chen, Huang, 2020]

Related Work (II)

Fixed-Order Page Number can be solved in $2^{O(\mathrm{vc}^3)}n$ time

[Bhore, Ganian, Montecchiani, Nöllenburg, 2020]

and in $2^{O(\mathrm{pw}^2)}n$ time, where pw is the pathwidth of the *ordered* graph, which is not bounded by vc.

[Liu, Chen, Huang, Wang, 2021]

Fixed-Order Book Drawing – testing if there is a p-page d-planar drawing of (G, σ) – can be solved in $(d+2)^{O(\mathrm{vc}^3)}n$ or in $(d+2)^{O(\mathrm{pw}^2)}n$ time. [Liu, Chen, Huang, 2020]

Bhore et al. [2020] also study the flexible vertex-order case: They solve Page Number in $2^{\mathrm{vc}^{O(\mathrm{vc})}} + \mathrm{vc} \log \mathrm{vc} \cdot n$ time.

Our Contribution

- We can compute the fixed-vertex-order page number of an ordered graph with m edges & n vertices in $2^m \cdot n^{O(1)}$ time. Alternatively, given a budget p of pages, we can compute a p-page book embedding with the min. number of crossings.

Our Contribution

- We can compute the fixed-vertex-order page number of an ordered graph with m edges & n vertices in $2^m \cdot n^{O(1)}$ time.
  Alternatively, given a budget p of pages, we can compute a p-page book embedding with the min. number of crossings.
- We obtain an $O((d+1)\log n)$-approximation algorithm for the fixed-vertex-order d-planar page number.

Our Contribution

- We can compute the fixed-vertex-order page number of an ordered graph with m edges & n vertices in $2^m \cdot n^{O(1)}$ time.
  Alternatively, given a budget p of pages, we can compute a p-page book embedding with the min. number of crossings.

- We obtain an $O((d+1)\log n)$-approximation algorithm for the fixed-vertex-order d-planar page number.

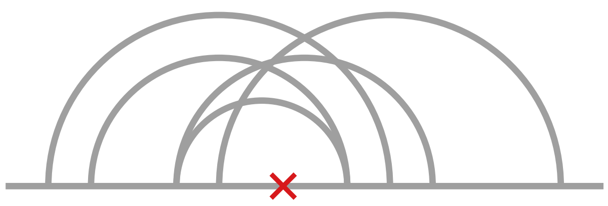
- We show how to decide in $2^{O(c\sqrt{k}\log(c+k))} \cdot n^{O(1)}$ time whether deleting k edges of an ordered graph suffices to obtain a d-planar layout on *one page*.

Our Contribution

- We can compute the fixed-vertex-order page number of an ordered graph with m edges & n vertices in $2^m \cdot n^{O(1)}$ time. Alternatively, given a budget p of pages, we can compute a p-page book embedding with the min. number of crossings.

- We obtain an $O((d+1)\log n)$-approximation algorithm for the fixed-vertex-order d-planar page number.

- We show how to decide in $2^{O(c\sqrt{k}\log(c+k))} \cdot n^{O(1)}$ time whether deleting k edges of an ordered graph suffices to obtain a d-planar layout on *one page*.



- Let h be the size of a *hitting set*.
  $h = 1$: We can *efficiently* compute the smallest set of edges whose deletion yields fixed-vertex-order page number p.
  $h > 1$: XP algorithm with respect to $h + p$.

Our Contribution

- We can compute the fixed-vertex-order page number of an ordered graph with m edges & n vertices in $2^m \cdot n^{O(1)}$ time.
  Alternatively, given a budget p of pages, we can compute a p-page book embedding with the min. number of crossings.
- We obtain an $O((d+1)\log n)$-approximation algorithm for the fixed-vertex-order d-planar page number.
- We show how to decide in $2^{O(c\sqrt{k}\log(c+k))} \cdot n^{O(1)}$ time whether deleting k edges of an ordered graph suffices to obtain a d-planar layout on *one page*.
- Let h be the size of a *hitting set*.
  $h = 1$: We can *efficiently* compute the smallest set of edges whose deletion yields fixed-vertex-order page number p.
  $h > 1$: XP algorithm with respect to $h + p$.

Minimizing Crossings (or Pages)

Liu, Chen, Huang, and Wang [2021] gave an algorithm that checks in $(\mathrm{cr}+2)^{O(\mathrm{pw}^2)} n$ time whether a graph with n vertices and pathwidth pw can be drawn on a given number of pages with $\leq$ cr crossings in total.

Minimizing Crossings (or Pages)

Liu, Chen, Huang, and Wang [2021] gave an algorithm that checks in $(\mathrm{cr}+2)^{O(\mathrm{pw}^2)}n$ time whether a graph with n vertices and pathwidth pw can be drawn on a given number of pages with $\leq$ cr crossings in total.

Given an ordered graph (G, σ), let $\mathrm{cr}_p(G, \sigma)$ be the smallest number of crossings over all possible assignments of the edges of G to p pages.

Minimizing Crossings (or Pages)

Liu, Chen, Huang, and Wang [2021] gave an algorithm that checks in $(\mathrm{cr}+2)^{O(\mathrm{pw}^2)}n$ time whether a graph with n vertices and pathwidth pw can be drawn on a given number of pages with $\leq$ cr crossings in total.

Given an ordered graph (G, σ), let $\mathrm{cr}_p(G, \sigma)$ be the smallest number of crossings over all possible assignments of the edges of G to p pages.

Theorem. Given $p \geq 1$ and an ordered graph (G, σ) with n vertices and m edges, we can compute the values $\mathrm{cr}_1(G, \sigma), \ldots, \mathrm{cr}_p(G, \sigma)$ in $2^m \cdot n^{O(1)}$ time.

Minimizing Crossings (or Pages)

Liu, Chen, Huang, and Wang [2021] gave an algorithm that checks in $(\mathrm{cr}+2)^{O(\mathrm{pw}^2)} n$ time whether a graph with n vertices and pathwidth pw can be drawn on a given number of pages with $\leq$ cr crossings in total.

Given an ordered graph (G, σ), let $\mathrm{cr}_p(G, \sigma)$ be the smallest number of crossings over all possible assignments of the edges of G to p pages.

Theorem. Given $p \geq 1$ and an ordered graph (G, σ) with n vertices and m edges, we can compute the values $\mathrm{cr}_1(G, \sigma), \ldots, \mathrm{cr}_p(G, \sigma)$ in $2^m \cdot n^{O(1)}$ time.

- In other words, given a budget p of pages, we can compute a p-page book embedding with the minimum number of crossings in ... time.

Minimizing Crossings (or Pages)

Liu, Chen, Huang, and Wang [2021] gave an algorithm that checks in $(\mathrm{cr}+2)^{O(\mathrm{pw}^2)} n$ time whether a graph with n vertices and pathwidth pw can be drawn on a given number of pages with $\leq$ cr crossings in total.

Given an ordered graph (G, σ), let $\mathrm{cr}_p(G, \sigma)$ be the smallest number of crossings over all possible assignments of the edges of G to p pages.

Theorem. Given $p \geq 1$ and an ordered graph (G, σ) with n vertices and m edges, we can compute the values $\mathrm{cr}_1(G, \sigma), \ldots, \mathrm{cr}_p(G, \sigma)$ in $2^m \cdot n^{O(1)}$ time.

- In other words, given a budget p of pages, we can compute a p-page book embedding with the minimum number of crossings in ... time.
- We can compute the *fixed-vertex-order page* number in ... time.

Minimizing Crossings (or Pages)

Liu, Chen, Huang, and Wang [2021] gave an algorithm that checks in $(\mathrm{cr}+2)^{O(\mathrm{pw}^2)} n$ time whether a graph with n vertices and pathwidth pw can be drawn on a given number of pages with $\leq$ cr crossings in total.

Given an ordered graph (G, σ), let $\mathrm{cr}_p(G, \sigma)$ be the smallest number of crossings over all possible assignments of the edges of G to p pages.

Theorem. Given $p \geq 1$ and an ordered graph (G, σ) with n vertices and m edges, we can compute the values $\mathrm{cr}_1(G, \sigma), \ldots, \mathrm{cr}_p(G, \sigma)$ in $2^m \cdot n^{O(1)}$ time.

- In other words, given a budget p of pages, we can compute a p-page book embedding with the minimum number of crossings in ... time.
- We can compute the *fixed-vertex-order page* number in ... time.
  Find the smallest q such that $\mathrm{cr}_q(G, \sigma) = 0$.

Minimizing Crossings (or Pages)

Liu, Chen, Huang, and Wang [2021] gave an algorithm that checks in $(\mathrm{cr}+2)^{O(\mathrm{pw}^2)} n$ time whether a graph with n vertices and pathwidth pw can be drawn on a given number of pages with $\leq$ cr crossings in total.

Given an ordered graph (G, σ), let $\mathrm{cr}_p(G, \sigma)$ be the smallest number of crossings over all possible assignments of the edges of G to p pages.

Theorem. Given $p \geq 1$ and an ordered graph (G, σ) with n vertices and m edges, we can compute the values $\mathrm{cr}_1(G, \sigma), \dots, \mathrm{cr}_p(G, \sigma)$ in $2^m \cdot n^{O(1)}$ time.

- In other words, given a budget p of pages, we can compute a p-page book embedding with the minimum number of crossings in . . . time.
- We can compute the *fixed-vertex-order page* number in . . . time.
  Find the smallest q such that $\mathrm{cr}_q(G, \sigma) = 0$. Note that $q \leq m$.

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathsf{cr}_1(G[F], \sigma) =$

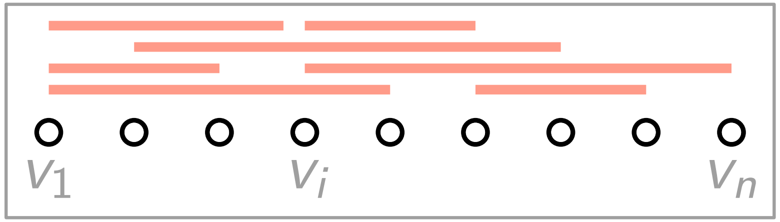
A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \Big|\Big\{\{e, f\} \subseteq F \colon e \text{ crosses } f\Big\}\Big|$.

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \left|\left\{\{e, f\} \subseteq F : e \text{ crosses } f\right\}\right|$.

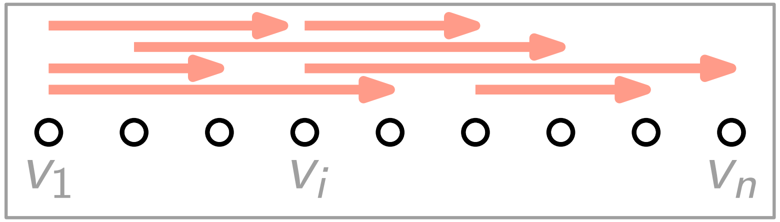
Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:



A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \left|\left\{\{e, f\} \subseteq F : e \text{ crosses } f\right\}\right|$.

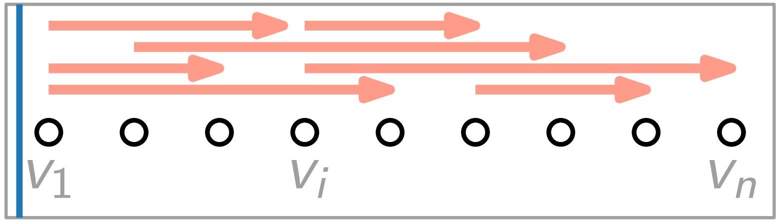
Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:



A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \big|\big\{\{e, f\} \subseteq F : e \text{ crosses } f\big\}\big|$.

Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:

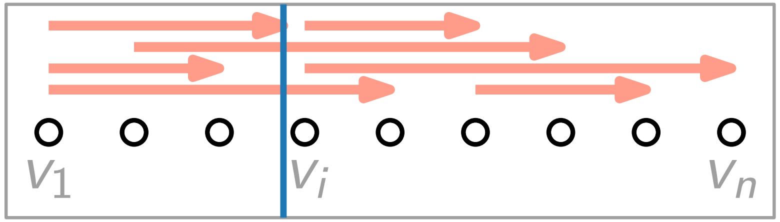


cr $= 0$; $B =$ empty BST for edges (right endpt)

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \left|\{\{e, f\} \subseteq F : e \text{ crosses } f\}\right|$.

Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:



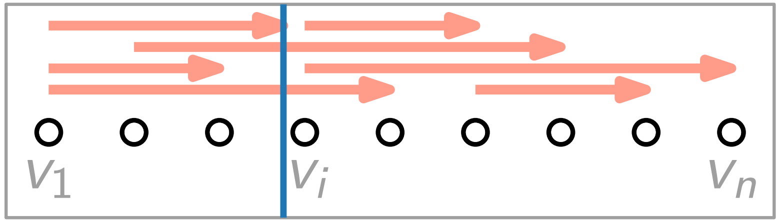
cr $= 0$; $B =$ empty BST for edges (right endpt)

For $i = 1$ to n:

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \left|\left\{\{e, f\} \subseteq F : e \text{ crosses } f\right\}\right|$.

Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:



cr $= 0$; $B =$ empty BST for edges (right endpt)

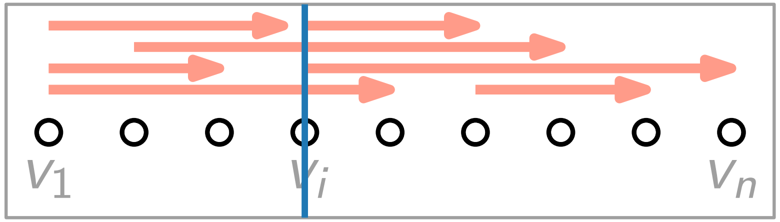
For $i = 1$ to n:

remove in(v_i) from B;

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\text{cr}_1(G[F], \sigma) = \left|\{\{e, f\} \subseteq F : e \text{ crosses } f\}\right|$.

Can compute $\text{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:



cr $= 0$; $B =$ empty BST for edges (right endpt)

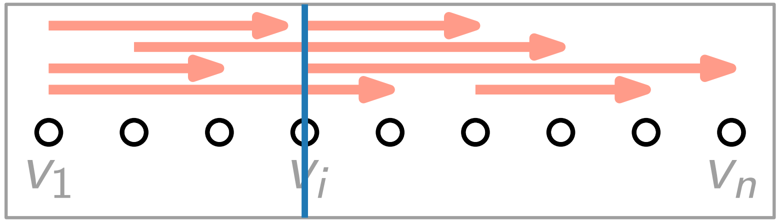
For $i = 1$ to n:

remove in(v_i) from B; for each $e \in$ out(v_i):

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \left|\left\{\{e, f\} \subseteq F : e \text{ crosses } f\right\}\right|$.

Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:



cr $= 0$; $B =$ empty BST for edges (right endpt)

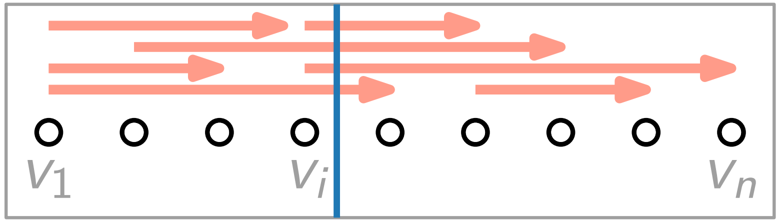
For $i = 1$ to n:

remove in(v_i) from B; for each $e \in$ out(v_i): cr $=$ cr $+ B$.rank(e);

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \left|\left\{\{e, f\} \subseteq F : e \text{ crosses } f\right\}\right|$.

Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:



cr $= 0$; $B =$ empty BST for edges (right endpt)

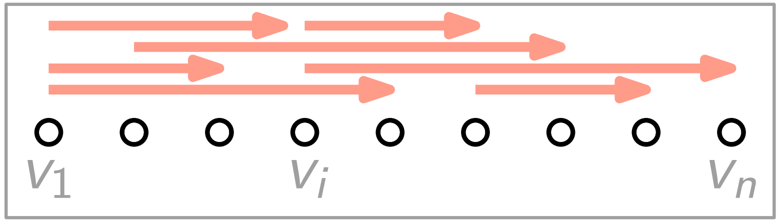
For $i = 1$ to n:

remove in(v_i) from B; for each $e \in$ out(v_i): cr $=$ cr $+ B$.rank(e); B.add(e)

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \left|\{\{e, f\} \subseteq F : e \text{ crosses } f\}\right|$.

Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:



cr = 0; B = empty BST for edges (right endpt)

For $i = 1$ to n:

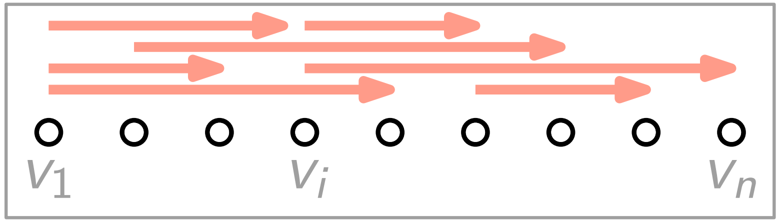
remove in(v_i) from B; for each $e \in$ out(v_i): cr = cr + B.rank(e); B.add(e)

So we can compute $\mathrm{cr}_1(G[F], \sigma)$ for all $F \subseteq E(G)$ in total time.

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \left|\{\{e, f\} \subseteq F : e \text{ crosses } f\}\right|$.

Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:



cr = 0; B = empty BST for edges (right endpt)

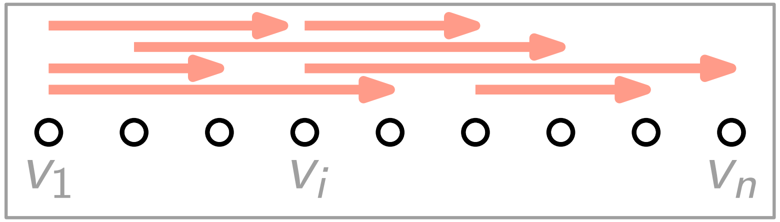
For $i = 1$ to n:

remove in(v_i) from B; for each $e \in$ out(v_i): cr = cr + B.rank(e); B.add(e)

So we can compute $\mathrm{cr}_1(G[F], \sigma)$ for all $F \subseteq E(G)$ in $\tilde{O}(m2^m)$ total time.

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \left|\left\{\{e, f\} \subseteq F : e \text{ crosses } f\right\}\right|$.



Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:

cr $= 0$; $B =$ empty BST for edges (right endpt)

For $i = 1$ to n:

remove $\mathrm{in}(v_i)$ from B; for each $e \in \mathrm{out}(v_i)$: cr $=$ cr $+ B.\mathrm{rank}(e)$; $B.\mathrm{add}(e)$

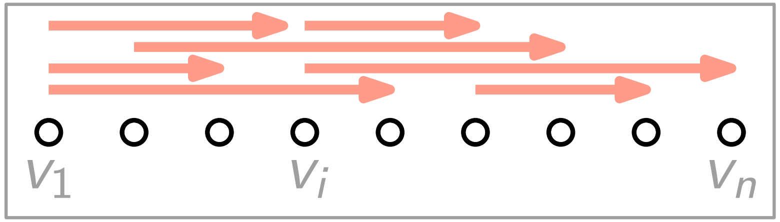
So we can compute $\mathrm{cr}_1(G[F], \sigma)$ for all $F \subseteq E(G)$ in $\tilde{O}(m2^m)$ total time.

For $q > 1$ and $F \subseteq E(G)$, we have the recurrence

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \left|\{\{e, f\} \subseteq F : e \text{ crosses } f\}\right|$.

Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:



cr $= 0$; B = empty BST for edges (right endpt)

For $i = 1$ to n:

remove in(v_i) from B; for each $e \in$ out(v_i): cr $=$ cr $+ B$.rank(e); B.add(e)

So we can compute $\mathrm{cr}_1(G[F], \sigma)$ for all $F \subseteq E(G)$ in $\tilde{O}(m2^m)$ total time.

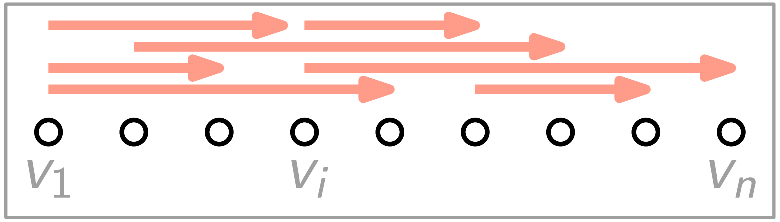
For $q > 1$ and $F \subseteq E(G)$, we have the recurrence

$$\mathrm{cr}_q(G[F], \sigma) = \min_{F' \subseteq F} \{ \qquad \}.$$

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \big|\{\{e, f\} \subseteq F : e \text{ crosses } f\}\big|$.

Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:



cr $= 0$; $B =$ empty BST for edges (right endpt)

For $i = 1$ to n:

remove in(v_i) from B; for each $e \in$ out(v_i): cr $=$ cr $+ B$.rank(e); B.add(e)

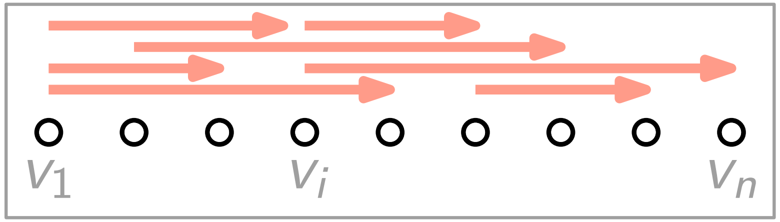
So we can compute $\mathrm{cr}_1(G[F], \sigma)$ for all $F \subseteq E(G)$ in $\tilde{O}(m2^m)$ total time.

For $q > 1$ and $F \subseteq E(G)$, we have the recurrence

$$\mathrm{cr}_q(G[F], \sigma) \;=\; \min_{F' \subseteq F} \left\{ \mathrm{cr}_1(G[F'], \sigma) + \mathrm{cr}_{q-1}(G[F \setminus F'], \sigma) \right\}.$$

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \big|\{\{e, f\} \subseteq F : e \text{ crosses } f\}\big|$.



Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:

cr $= 0$; $B =$ empty BST for edges (right endpt)

For $i = 1$ to n:

remove in(v_i) from B; for each $e \in$ out(v_i): cr $=$ cr $+ B$.rank(e); B.add(e)

So we can compute $\mathrm{cr}_1(G[F], \sigma)$ for all $F \subseteq E(G)$ in $\tilde{O}(m2^m)$ total time.

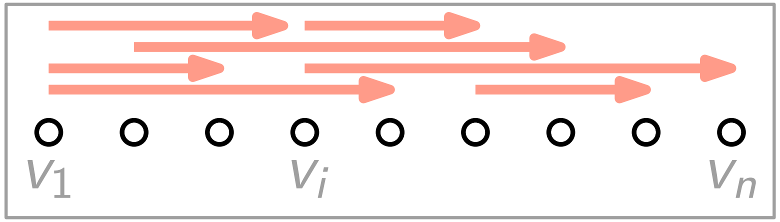
For $q > 1$ and $F \subseteq E(G)$, we have the recurrence

$$\mathrm{cr}_q(G[F], \sigma) \;=\; \min_{F' \subseteq F} \{\, \mathrm{cr}_1(G[F'], \sigma) + \mathrm{cr}_{q-1}(G[F \setminus F'], \sigma) \,\}.$$

Brute-force computation takes time $O(m) \cdot \sum_{i=1}^{m} \binom{m}{i} 2^i =$

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \big|\{\{e, f\} \subseteq F : e \text{ crosses } f\}\big|$.



Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:

cr $= 0$; $B =$ empty BST for edges (right endpt)

For $i = 1$ to n:

remove in(v_i) from B; for each $e \in$ out(v_i): cr $=$ cr $+ B$.rank(e); B.add(e)

So we can compute $\mathrm{cr}_1(G[F], \sigma)$ for all $F \subseteq E(G)$ in $\tilde{O}(m2^m)$ total time.

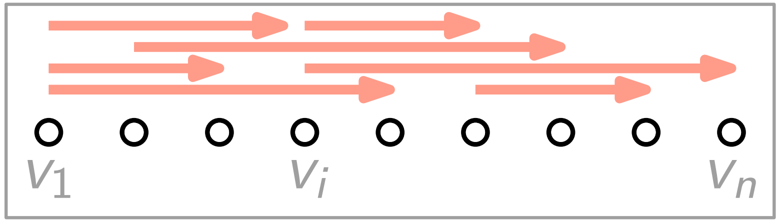
For $q > 1$ and $F \subseteq E(G)$, we have the recurrence

$$\mathrm{cr}_q(G[F], \sigma) \;=\; \min_{F' \subseteq F} \left\{ \mathrm{cr}_1(G[F'], \sigma) + \mathrm{cr}_{q-1}(G[F \setminus F'], \sigma) \right\}.$$

Brute-force computation takes time $O(m) \cdot \sum_{i=1}^{m} \binom{m}{i} 2^i = \tilde{O}(m3^m)$.

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \left|\left\{\{e, f\} \subseteq F : e \text{ crosses } f\right\}\right|$.



Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:

$\mathrm{cr} = 0$; $B =$ empty BST for edges (right endpt)

For $i = 1$ to n:

remove in(v_i) from B; for each $e \in \mathrm{out}(v_i)$: $\mathrm{cr} = \mathrm{cr} + B.\mathrm{rank}(e)$; $B.\mathrm{add}(e)$

So we can compute $\mathrm{cr}_1(G[F], \sigma)$ for all $F \subseteq E(G)$ in $\tilde{O}(m2^m)$ total time.

For $q > 1$ and $F \subseteq E(G)$, we have the recurrence

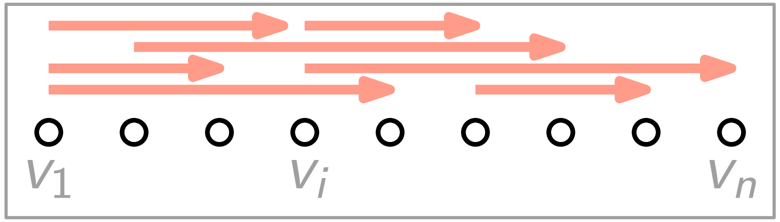
$$\mathrm{cr}_q(G[F], \sigma) \;=\; \min_{F' \subseteq F} \left\{ \mathrm{cr}_1(G[F'], \sigma) + \mathrm{cr}_{q-1}(G[F \setminus F'], \sigma) \right\}.$$

Brute-force computation takes time $O(m) \cdot \sum_{i=1}^{m} \binom{m}{i} 2^i = \tilde{O}(m3^m)$.

Instead, do *subset convolution*! [Björklund, Husfeldt, Kaski, Koivisto, '07]

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\text{cr}_1(G[F], \sigma) = \big|\big\{\{e, f\} \subseteq F : e \text{ crosses } f\big\}\big|$.



Can compute $\text{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:

cr $= 0$; $B =$ empty BST for edges (right endpt)

For $i = 1$ to n:

remove in(v_i) from B; for each $e \in$ out(v_i): cr $=$ cr $+ B$.rank(e); B.add(e)

So we can compute $\text{cr}_1(G[F], \sigma)$ for all $F \subseteq E(G)$ in $\tilde{O}(m2^m)$ total time.

For $q > 1$ and $F \subseteq E(G)$, we have the recurrence

$$\text{cr}_q(G[F], \sigma) = \min_{F' \subseteq F} \left\{ \text{cr}_1(G[F'], \sigma) + \text{cr}_{q-1}(G[F \setminus F'], \sigma) \right\}.$$

Brute-force computation takes time $O(m) \cdot \sum_{i=1}^{m} \binom{m}{i} 2^i = \tilde{O}(m3^m)$.

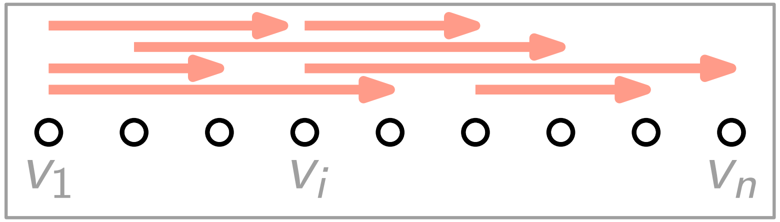
Instead, do *subset convolution*! [Björklund, Husfeldt, Kaski, Koivisto, '07]

Define two functions $f, g \colon 2^{E(G)} \to R$ to the $(\min, +)$-ring R:

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \big|\{\{e,f\} \subseteq F \colon e \text{ crosses } f\}\big|$.

Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:



cr = 0; B = empty BST for edges (right endpt)

For $i = 1$ to n:

remove in(v_i) from B; for each $e \in$ out(v_i): cr = cr + B.rank(e); B.add(e)

So we can compute $\mathrm{cr}_1(G[F], \sigma)$ for all $F \subseteq E(G)$ in $\tilde{O}(m2^m)$ total time.

For $q > 1$ and $F \subseteq E(G)$, we have the recurrence

$$\mathrm{cr}_q(G[F], \sigma) \;=\; \min_{F' \subseteq F} \big\{\, \mathrm{cr}_1(G[F'], \sigma) + \mathrm{cr}_{q-1}(G[F \setminus F'], \sigma) \,\big\}.$$

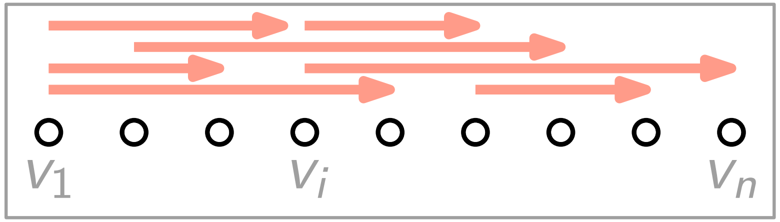
Brute-force computation takes time $O(m) \cdot \sum_{i=1}^{m} \binom{m}{i} 2^i = \tilde{O}(m3^m)$.

Instead, do *subset convolution*! [Björklund, Husfeldt, Kaski, Koivisto, '07]

Define two functions $f, g \colon 2^{E(G)} \to R$ to the (min, +)-ring R:

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \big|\{\{e, f\} \subseteq F : e \text{ crosses } f\}\big|$.



Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:

cr $= 0$; $B =$ empty BST for edges (right endpt)

For $i = 1$ to n:

remove in(v_i) from B; for each $e \in$ out(v_i): cr $=$ cr $+ B$.rank(e); B.add(e)

So we can compute $\mathrm{cr}_1(G[F], \sigma)$ for all $F \subseteq E(G)$ in $\tilde{O}(m2^m)$ total time.

For $q > 1$ and $F \subseteq E(G)$, we have the recurrence

$$\mathrm{cr}_q(G[F], \sigma) \;=\; \min_{F' \subseteq F} \left\{ \mathrm{cr}_1(G[F'], \sigma) + \mathrm{cr}_{q-1}(G[F \setminus F'], \sigma) \right\}.$$

Brute-force computation takes time $O(m) \cdot \sum_{i=1}^{m} \binom{m}{i} 2^i = \tilde{O}(m3^m)$.

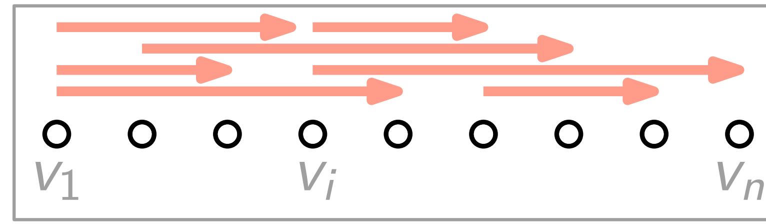
Instead, do *subset convolution*! [Björklund, Husfeldt, Kaski, Koivisto, '07]

Define two functions $f, g \colon 2^{E(G)} \to R$ to the $(\min, +)$-ring R:

$f \colon F \mapsto \mathrm{cr}_1(G[F], \sigma)$ and $g \colon F \mapsto \mathrm{cr}_{q-1}(G[F], \sigma)$ with $f, g(\cdot) \leq m^2$

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \big|\{\{e, f\} \subseteq F : e \text{ crosses } f\}\big|$.



Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:

$\mathrm{cr} = 0$; B = empty BST for edges (right endpt)

For $i = 1$ to n:

remove $\mathrm{in}(v_i)$ from B; for each $e \in \mathrm{out}(v_i)$: $\mathrm{cr} = \mathrm{cr} + B.\mathrm{rank}(e)$; $B.\mathrm{add}(e)$

So we can compute $\mathrm{cr}_1(G[F], \sigma)$ for all $F \subseteq E(G)$ in $\tilde{O}(m2^m)$ total time.

For $q > 1$ and $F \subseteq E(G)$, we have the recurrence

$$\mathrm{cr}_q(G[F], \sigma) \; = \; \min_{F' \subseteq F} \left\{ \mathrm{cr}_1(G[F'], \sigma) + \mathrm{cr}_{q-1}(G[F \setminus F'], \sigma) \right\}.$$

Brute-force computation takes time $O(m) \cdot \sum_{i=1}^{m} \binom{m}{i} 2^i = \tilde{O}(m3^m)$.

Instead, do *subset convolution*! [Björklund, Husfeldt, Kaski, Koivisto, '07]

Define two functions $f, g \colon 2^{E(G)} \to R$ to the $(\min, +)$-ring R:

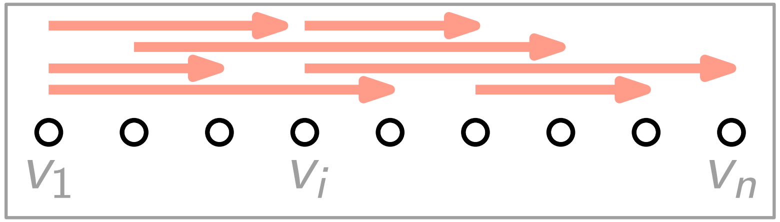
$f \colon F \mapsto \mathrm{cr}_1(G[F], \sigma)$ and $g \colon F \mapsto \mathrm{cr}_{q-1}(G[F], \sigma)$ with $f, g(\cdot) \leq m^2$

Then $(f * g)(F) = \sum_{F' \subseteq F} f(F') \cdot g(F \setminus F')$ can be computed in $\tilde{O}(m^2 2^m)$.

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \left|\{\{e, f\} \subseteq F : e \text{ crosses } f\}\right|$.

Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:



cr $= 0$; $B =$ empty BST for edges (right endpt)

For $i = 1$ to n:

remove $\mathrm{in}(v_i)$ from B; for each $e \in \mathrm{out}(v_i)$: cr $=$ cr $+ B.\mathrm{rank}(e)$; $B.\mathrm{add}(e)$

So we can compute $\mathrm{cr}_1(G[F], \sigma)$ for all $F \subseteq E(G)$ in $\tilde{O}(m2^m)$ total time.

For $q > 1$ and $F \subseteq E(G)$, we have the recurrence

$$\mathrm{cr}_q(G[F], \sigma) \;=\; \min_{F' \subseteq F} \left\{ \mathrm{cr}_1(G[F'], \sigma) + \mathrm{cr}_{q-1}(G[F \setminus F'], \sigma) \right\}.$$

Brute-force computation takes time $O(m) \cdot \sum_{i=1}^{m} \binom{m}{i} 2^i = \tilde{O}(m3^m)$.

Instead, do *subset convolution*! [Björklund, Husfeldt, Kaski, Koivisto, '07]

Define two functions $f, g \colon 2^{E(G)} \to R$ to the $(\min, +)$-ring R:

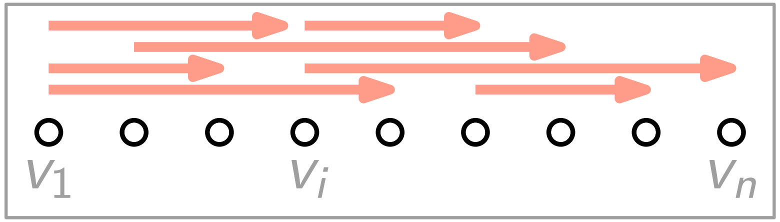
$f \colon F \mapsto \mathrm{cr}_1(G[F], \sigma)$ and $g \colon F \mapsto \mathrm{cr}_{q-1}(G[F], \sigma)$ with $f, g(\cdot) \leq m^2$

Then $(f * g)(F) = \sum_{F' \subseteq F} f(F') \cdot g(F \setminus F')$ can be computed in $\tilde{O}(m^2 2^m)$.

A Nice Tool: Subset Convolution

Let $p = 1$ and $F \subseteq E(G)$. Then $\mathrm{cr}_1(G[F], \sigma) = \big|\{\{e, f\} \subseteq F : e \text{ crosses } f\}\big|$.

Can compute $\mathrm{cr}_1(G[F], \sigma)$ in $\tilde{O}(|F|)$ time:



cr $= 0$; $B =$ empty BST for edges (right endpt)

For $i = 1$ to n:

remove in(v_i) from B; for each $e \in$ out(v_i): cr = cr $+ B$.rank(e); B.add(e)

So we can compute $\mathrm{cr}_1(G[F], \sigma)$ for all $F \subseteq E(G)$ in $\tilde{O}(m2^m)$ total time.

For $q > 1$ and $F \subseteq E(G)$, we have the recurrence

$$\mathrm{cr}_q(G[F], \sigma) \;=\; \min_{F' \subseteq F} \left\{ \mathrm{cr}_1(G[F'], \sigma) + \mathrm{cr}_{q-1}(G[F \setminus F'], \sigma) \right\}.$$

Brute-force computation takes time $O(m) \cdot \sum_{i=1}^{m} \binom{m}{i} 2^i = \tilde{O}(m3^m)$.

Instead, do *subset convolution*! [Björklund, Husfeldt, Kaski, Koivisto, '07]

Define two functions $f, g \colon 2^{E(G)} \to R$ to the $(\min, +)$-ring R:

$f \colon F \mapsto \mathrm{cr}_1(G[F], \sigma)$ and $g \colon F \mapsto \mathrm{cr}_{q-1}(G[F], \sigma)$ with $f, g(\cdot) \leq m^2$

Then $(f * g)(F) = \sum_{F' \subseteq F} f(F') \cdot g(F \setminus F')$ can be computed in $\tilde{O}(m^2 2^m)$.

A Nice Tool: Subset Convolution

Theorem. Given $p \geq 1$ and an ordered graph (G, σ) with n vertices and m edges, we can compute the values $\mathrm{cr}_1(G, \sigma), \ldots, \mathrm{cr}_p(G, \sigma)$ in $\tilde{O}(p \cdot m^2 2^m)$ time.

So we can compute $\mathrm{cr}_1(G[F], \sigma)$ for all $F \subseteq E(G)$ in $\tilde{O}(m2^m)$ total time.

For $q > 1$ and $F \subseteq E(G)$, we have the recurrence

$$\mathrm{cr}_q(G[F], \sigma) \; = \; \min_{F' \subseteq F} \{ \mathrm{cr}_1(G[F'], \sigma) + \mathrm{cr}_{q-1}(G[F \setminus F'], \sigma) \}.$$

Brute-force computation takes time $O(m) \cdot \sum_{i=1}^{m} \binom{m}{i} 2^i = \tilde{O}(m3^m)$.

Instead, do *subset convolution*! [Björklund, Husfeldt, Kaski, Koivisto, '07]

Define two functions $f, g \colon 2^{E(G)} \to R$ to the $(\min, +)$-ring R:

$f \colon F \mapsto \mathrm{cr}_1(G[F], \sigma)$ and $g \colon F \mapsto \mathrm{cr}_{q-1}(G[F], \sigma)$ with $f, g(\cdot) \leq m^2$

Then $(f * g)(F) = \sum_{F' \subseteq F} f(F') \cdot g(F \setminus F')$ can be computed in $\tilde{O}(m^2 2^m)$.

Need the Fixed-Order Page Number Faster?

Need the Fixed-Order Page Number Faster?

Lemma. Given (G, σ), we can compute in quadratic time a smallest set $S \subseteq E(G)$ such that $\mathrm{cr}_1(G - S, \sigma) = 0$.

Need the Fixed-Order Page Number Faster?

Lemma. Given (G, σ), we can compute in quadratic time a smallest set $S \subseteq E(G)$ such that $\mathrm{cr}_1(G - S, \sigma) = 0$.

Via max. indep. set in circle graphs. [Valiente 2003]

Need the Fixed-Order Page Number Faster?

Lemma. Given (G, σ), we can compute in quadratic time a smallest set $S \subseteq E(G)$ such that $\mathrm{cr}_1(G - S, \sigma) = 0$.

Via max. indep. set in circle graphs. [Valiente 2003]

Theorem. We can compute an $O(\log n)$-approximation to the fixed-vertex-order page number of an n-vertex graph.

Need the Fixed-Order Page Number Faster?

Lemma. Given (G, σ), we can compute in quadratic time a smallest set $S \subseteq E(G)$ such that $\mathrm{cr}_1(G - S, \sigma) = 0$.
Via max. indep. set in circle graphs. [Valiente 2003]

Theorem. We can compute an $O(\log n)$-approximation to the fixed-vertex-order page number of an n-vertex graph.

Proof. Let $\mathcal{F} = \{F \subseteq E(G) \colon \mathrm{cr}_1(G[F], \sigma) = 0\}$.

Need the Fixed-Order Page Number Faster?

Lemma. Given (G, σ), we can compute in quadratic time a smallest set $S \subseteq E(G)$ such that $\mathrm{cr}_1(G - S, \sigma) = 0$.
Via max. indep. set in circle graphs. [Valiente 2003]

Theorem. We can compute an $O(\log n)$-approximation to the fixed-vertex-order page number of an n-vertex graph.

Proof. Let $\mathcal{F} = \{F \subseteq E(G) \colon \mathrm{cr}_1(G[F], \sigma) = 0\}$.

$\mathcal{F}' \subseteq \mathcal{F}$ is a feasible solution of the Set Cover instance $(E(G), \mathcal{F})$ if $\bigcup \mathcal{F}' = E(G)$.

Need the Fixed-Order Page Number Faster?

Lemma. Given (G, σ), we can compute in quadratic time a smallest set $S \subseteq E(G)$ such that $\mathrm{cr}_1(G - S, \sigma) = 0$.
Via max. indep. set in circle graphs. [Valiente 2003]

Theorem. We can compute an $O(\log n)$-approximation to the fixed-vertex-order page number of an n-vertex graph.

Proof. Let $\mathcal{F} = \{F \subseteq E(G) \colon \mathrm{cr}_1(G[F], \sigma) = 0\}$.

$\mathcal{F}' \subseteq \mathcal{F}$ is a feasible solution of the Set Cover instance $(E(G), \mathcal{F})$ if $\bigcup \mathcal{F}' = E(G)$.

$\mathcal{F}'$ yields a crossing-free drawing of G on $|\mathcal{F}'|$ pages.

Need the Fixed-Order Page Number Faster?

Lemma. Given (G, σ), we can compute in quadratic time a smallest set $S \subseteq E(G)$ such that $\mathrm{cr}_1(G - S, \sigma) = 0$.
Via max. indep. set in circle graphs. [Valiente 2003]

Theorem. We can compute an $O(\log n)$-approximation to the fixed-vertex-order page number of an n-vertex graph.

Proof. Let $\mathcal{F} = \{F \subseteq E(G) \colon \mathrm{cr}_1(G[F], \sigma) = 0\}$.

$\mathcal{F}' \subseteq \mathcal{F}$ is a feasible solution of the Set Cover instance $(E(G), \mathcal{F})$ if $\bigcup \mathcal{F}' = E(G)$.

$\mathcal{F}'$ yields a crossing-free drawing of G on $|\mathcal{F}'|$ pages.

An opt/app. set cover yields an opt/app. page nmb.

Need the Fixed-Order Page Number Faster?

Lemma. Given (G, σ), we can compute in quadratic time a smallest set $S \subseteq E(G)$ such that $\mathrm{cr}_1(G - S, \sigma) = 0$.
Via max. indep. set in circle graphs. [Valiente 2003]

Theorem. We can compute an $O(\log n)$-approximation to the fixed-vertex-order page number of an n-vertex graph.

Proof. Let $\mathcal{F} = \{F \subseteq E(G) \colon \mathrm{cr}_1(G[F], \sigma) = 0\}$.

$\mathcal{F}' \subseteq \mathcal{F}$ is a feasible solution of the Set Cover instance $(E(G), \mathcal{F})$ if $\bigcup \mathcal{F}' = E(G)$.

$\mathcal{F}'$ yields a crossing-free drawing of G on $|\mathcal{F}'|$ pages.

An opt/app. set cover yields an opt/app. page nmb.

Compute solution $\mathcal{S}$ by greedily adding the set F in $\mathcal{F}$ that maximizes $|F \setminus \bigcup \mathcal{S}|$.

Need the Fixed-Order Page Number Faster?

Lemma. Given (G, σ), we can compute in quadratic time a smallest set $S \subseteq E(G)$ such that $\mathrm{cr}_1(G - S, \sigma) = 0$.
Via max. indep. set in circle graphs. [Valiente 2003]

Theorem. We can compute an $O(\log n)$-approximation to the fixed-vertex-order page number of an n-vertex graph.

Proof. Let $\mathcal{F} = \{F \subseteq E(G) \colon \mathrm{cr}_1(G[F], \sigma) = 0\}$.

$\mathcal{F}' \subseteq \mathcal{F}$ is a feasible solution of the Set Cover instance $(E(G), \mathcal{F})$ if $\bigcup \mathcal{F}' = E(G)$.

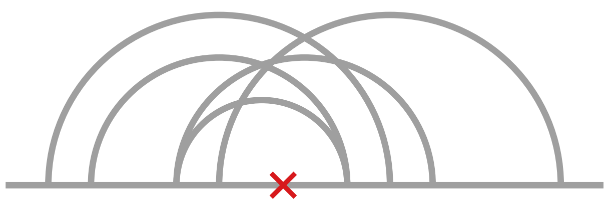
$\mathcal{F}'$ yields a crossing-free drawing of G on $|\mathcal{F}'|$ pages.

An opt/app. set cover yields an opt/app. page nmb.

Compute solution $\mathcal{S}$ by greedily adding the set F in $\mathcal{F}$ that maximizes $|F \setminus \bigcup \mathcal{S}|$. Apply lemma to $G - \bigcup \mathcal{S}$.

Our Contribution

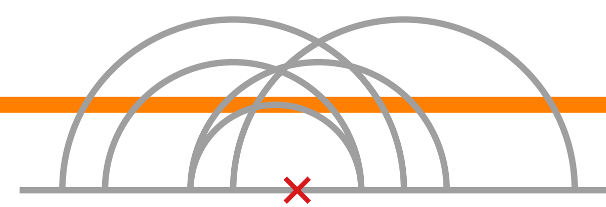
- We can compute the fixed-vertex-order page number of an ordered graph with m edges & n vertices in $2^m \cdot n^{O(1)}$ time.
  Alternatively, given a budget p of pages, we can compute a p-page book embedding with the min. number of crossings.
- We obtain an $O((d+1)\log n)$-approximation algorithm for the fixed-vertex-order d-planar page number.
- We show how to decide in $2^{O(c\sqrt{k}\log(c+k))} \cdot n^{O(1)}$ time whether deleting k edges of an ordered graph suffices to obtain a d-planar layout on *one* page.



- Let h be the size of a *hitting set*.
  $h = 1$: We can *efficiently* compute the smallest set of edges whose deletion yields fixed-vertex-order page number p.
  $h > 1$: XP algorithm with respect to $h + p$.

Our Contribution

- We can compute the fixed-vertex-order page number of an ordered graph with m edges & n vertices in $2^m \cdot n^{O(1)}$ time.
  Alternatively, given a budget p of pages, we can compute a p-page book embedding with the min. number of crossings.
- We obtain an $O((d+1)\log n)$-approximation algorithm for the fixed-vertex-order d-planar page number.
- We show how to decide in $2^{O(c\sqrt{k}\log(c+k))} \cdot n^{O(1)}$ time whether deleting k edges of an ordered graph suffices to obtain a d-planar layout on *one* page.



- Let h be the size of a *hitting set*.
  $h = 1$: We can *efficiently* compute the smallest set of edges whose deletion yields fixed-vertex-order page number p.
  $h > 1$: XP algorithm with respect to $h + p$.

Edge Deletion to p-Page Planar

no crossings

Brute-force solution?

Edge Deletion to p-Page Planar

no crossings

Brute-force solution?

For each mapping of the m edges to the p pages (allowing also for edge deletion), check for each page whether the edges assigned to it form an outerplanar graph.

Edge Deletion to p-Page Planar

no crossings

Brute-force solution?

For each mapping of the m edges to the p pages (allowing also for edge deletion), check for each page whether the edges assigned to it form an outerplanar graph. $\rightarrow O(n(p+1)^m)$ time

Edge Deletion to p-Page Planar

no crossings

Brute-force solution?

For each mapping of the m edges to the p pages (allowing also for edge deletion), check for each page whether the edges assigned to it form an outerplanar graph. $\rightarrow O(n(p+1)^m)$ time

New parameter:

Edge Deletion to p-Page Planar

no crossings

Brute-force solution?

For each mapping of the m edges to the p pages (allowing also for edge deletion), check for each page whether the edges assigned to it form an outerplanar graph. $\rightarrow O(n(p+1)^m)$ time

New parameter: h = size of hitting set

Edge Deletion to p-Page Planar

no crossings

Brute-force solution?

For each mapping of the m edges to the p pages (allowing also for edge deletion), check for each page whether the edges assigned to it form an outerplanar graph. $\rightarrow O(n(p+1)^m)$ time

New parameter: h = size of hitting set

- A hitting set can be much smaller than a vertex cover :-)

Edge Deletion to p-Page Planar

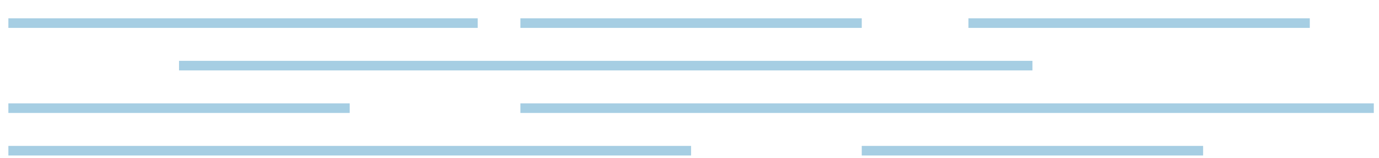
no crossings

Brute-force solution?

For each mapping of the m edges to the p pages (allowing also for edge deletion), check for each page whether the edges assigned to it form an outerplanar graph. $\rightarrow O(n(p+1)^m)$ time

New parameter: h = size of hitting set

- A hitting set can be much smaller than a vertex cover :-)
- Given m open intervals, a minimum-size hitting set can be found in $O(m \log m)$ time (greedily).



Edge Deletion to p-Page Planar

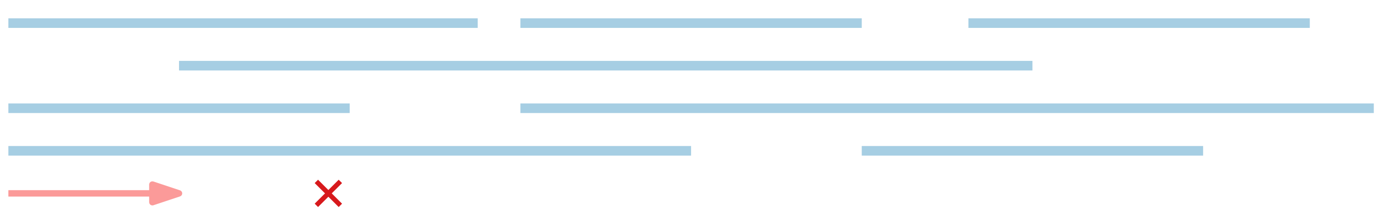
no crossings

Brute-force solution?

For each mapping of the m edges to the p pages (allowing also for edge deletion), check for each page whether the edges assigned to it form an outerplanar graph. $\rightarrow O(n(p+1)^m)$ time

New parameter: $h =$ size of hitting set

- A hitting set can be much smaller than a vertex cover :-)
- Given m open intervals, a minimum-size hitting set can be found in $O(m \log m)$ time (greedily).



Edge Deletion to p-Page Planar

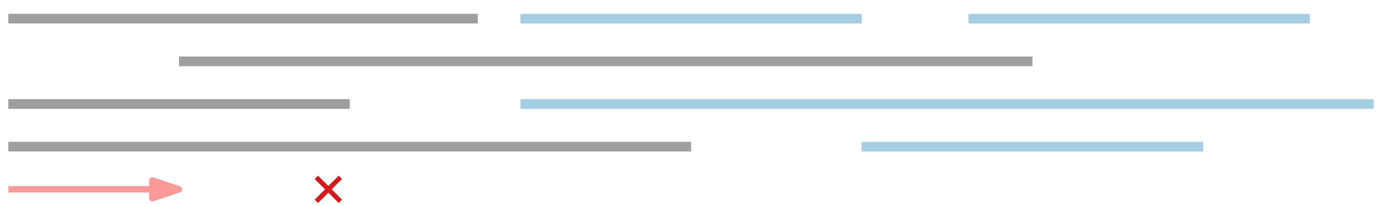
no crossings

Brute-force solution?

For each mapping of the m edges to the p pages (allowing also for edge deletion), check for each page whether the edges assigned to it form an outerplanar graph. $\rightarrow O(n(p+1)^m)$ time

New parameter: $h =$ size of hitting set

- A hitting set can be much smaller than a vertex cover :-)
- Given m open intervals, a minimum-size hitting set can be found in $O(m \log m)$ time (greedily).



Edge Deletion to p-Page Planar

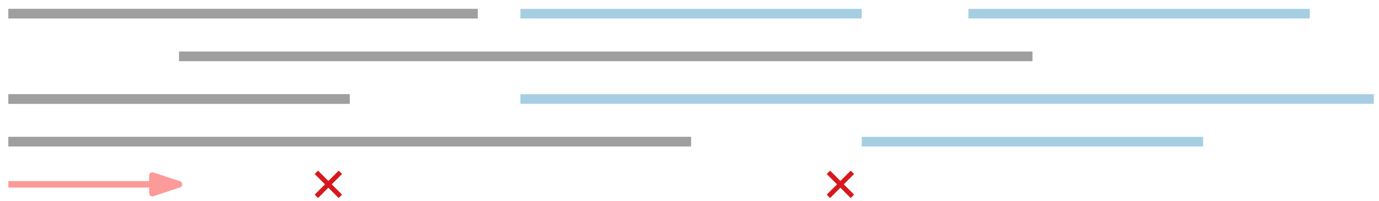
no crossings

Brute-force solution?

For each mapping of the m edges to the p pages (allowing also for edge deletion), check for each page whether the edges assigned to it form an outerplanar graph. $\rightarrow O(n(p+1)^m)$ time

New parameter: $h =$ size of hitting set

- A hitting set can be much smaller than a vertex cover :-)
- Given m open intervals, a minimum-size hitting set can be found in $O(m \log m)$ time (greedily).



Edge Deletion to p-Page Planar

no crossings

Brute-force solution?

For each mapping of the m edges to the p pages (allowing also for edge deletion), check for each page whether the edges assigned to it form an outerplanar graph. $\rightarrow O(n(p+1)^m)$ time

New parameter: h = size of hitting set

- A hitting set can be much smaller than a vertex cover :-)
- Given m open intervals, a minimum-size hitting set can be found in $O(m \log m)$ time (greedily).

Edge Deletion to p-Page Planar

no crossings

Brute-force solution?

For each mapping of the m edges to the p pages (allowing also for edge deletion), check for each page whether the edges assigned to it form an outerplanar graph. $\rightarrow O(n(p+1)^m)$ time

New parameter: h = size of hitting set

- A hitting set can be much smaller than a vertex cover :-)
- Given m open intervals, a minimum-size hitting set can be found in $O(m \log m)$ time (greedily).

EDGE DELETION TO p-PAGE PLANAR

no crossings

Brute-force solution?

For each mapping of the m edges to the p pages (allowing also for edge deletion), check for each page whether the edges assigned to it form an outerplanar graph. $\rightarrow O(n(p+1)^m)$ time

New parameter: $h =$ size of hitting set

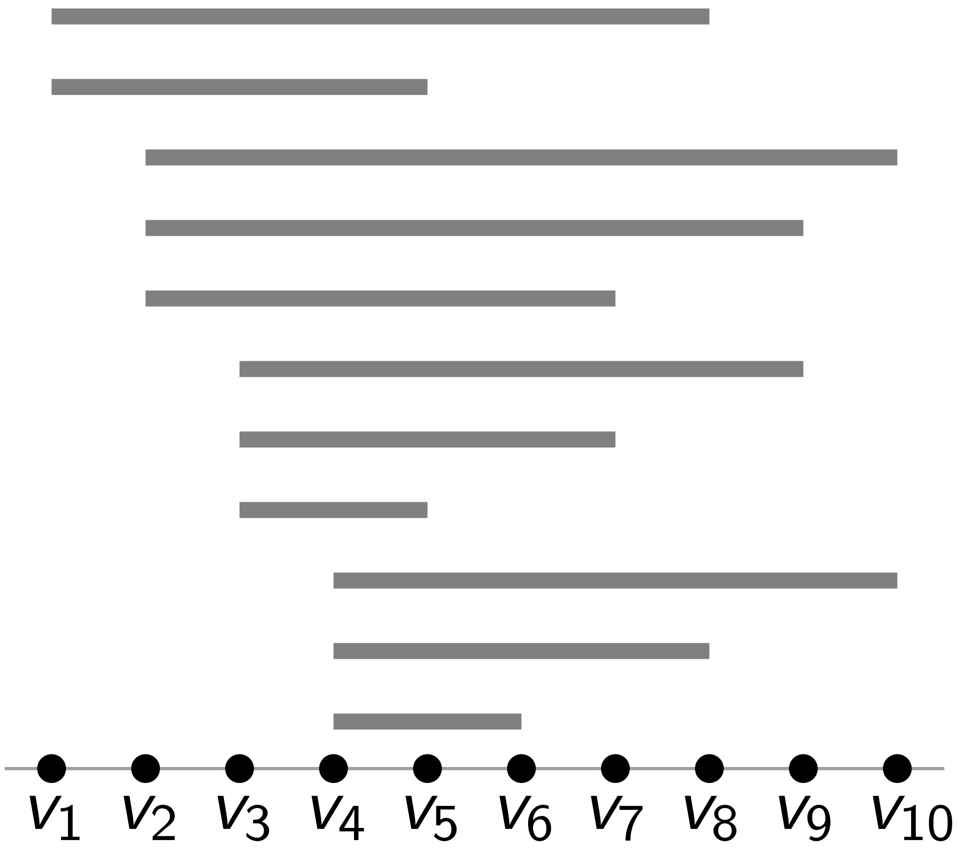
- A hitting set can be much smaller than a vertex cover :-)
- Given m open intervals, a minimum-size hitting set can be found in $O(m \log m)$ time (greedily).

Hitting Set of Size 1

Theorem. Given an ordered graph (G, σ) with n vertices, m edges, and $h(G, \sigma) = 1$, Edge Deletion to p-Page Planar can be solved in $O(m^3 \log n \log \log p)$ time.

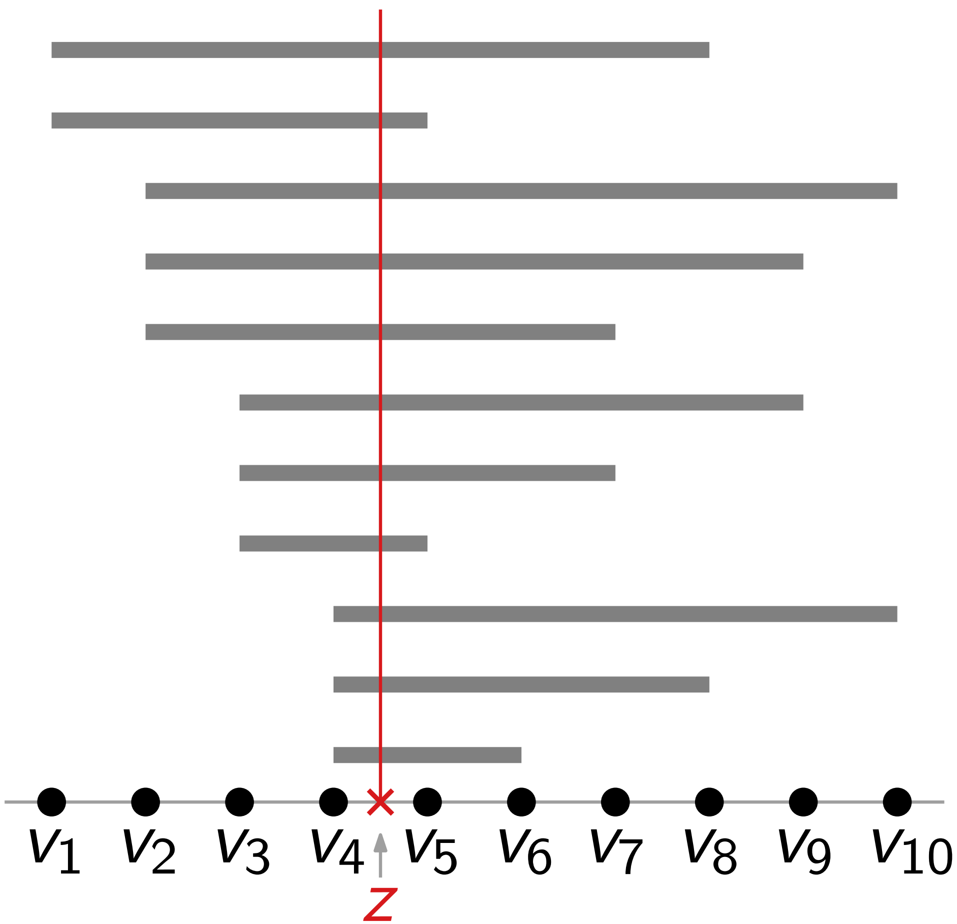
Hitting Set of Size 1

Theorem. Given an ordered graph (G, σ) with n vertices, m edges, and $h(G, \sigma) = 1$, Edge Deletion to p-Page Planar can be solved in $O(m^3 \log n \log \log p)$ time.



Hitting Set of Size 1

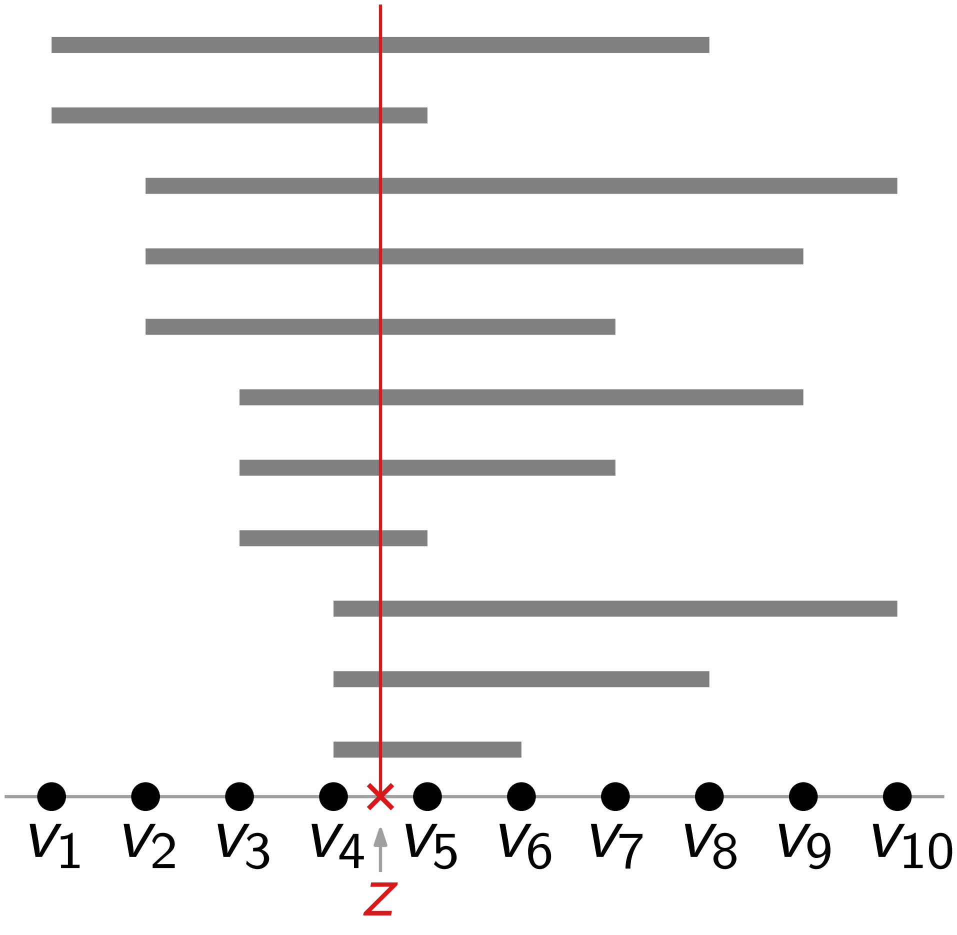
Theorem. Given an ordered graph (G, σ) with n vertices, m edges, and $h(G, \sigma) = 1$, Edge Deletion to p-Page Planar can be solved in $O(m^3 \log n \log\log p)$ time.



Hitting Set of Size 1

Theorem. Given an ordered graph (G, σ) with n vertices, m edges, and $h(G, \sigma) = 1$, Edge Deletion to p-Page Planar can be solved in $O(m^3 \log n \log\log p)$ time.

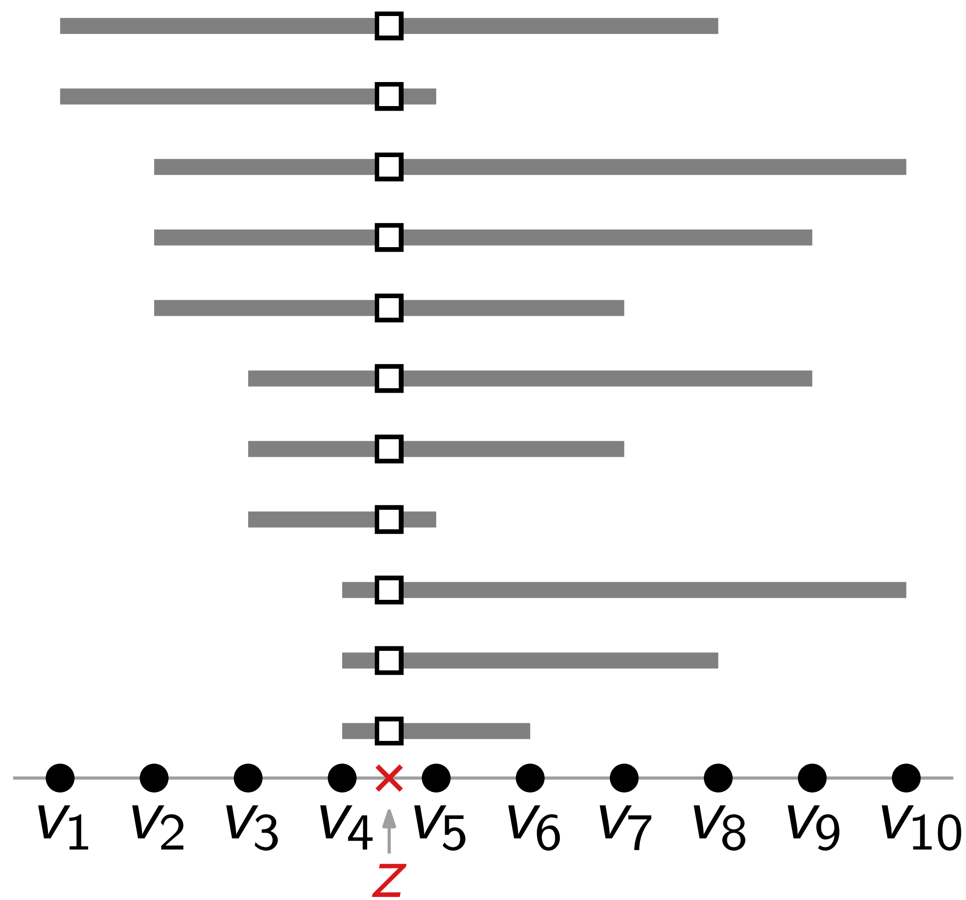
Proof.



Hitting Set of Size 1

Theorem. Given an ordered graph (G, σ) with n vertices, m edges, and $h(G, \sigma) = 1$, Edge Deletion to p-Page Planar can be solved in $O(m^3 \log n \log \log p)$ time.

Proof.

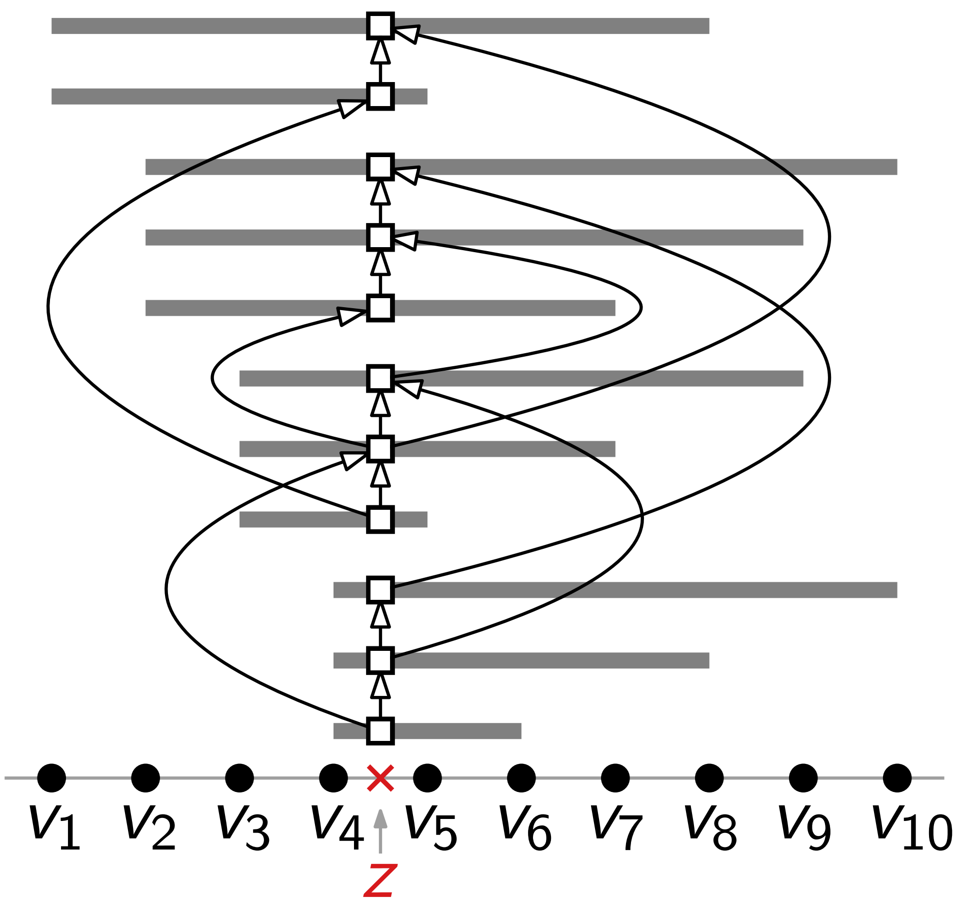


Hitting Set of Size 1

Theorem. Given an ordered graph (G, σ) with n vertices, m edges, and $h(G, \sigma) = 1$, Edge Deletion to p-Page Planar can be solved in $O(m^3 \log n \log\log p)$ time.

Proof.

– Define directed graph.

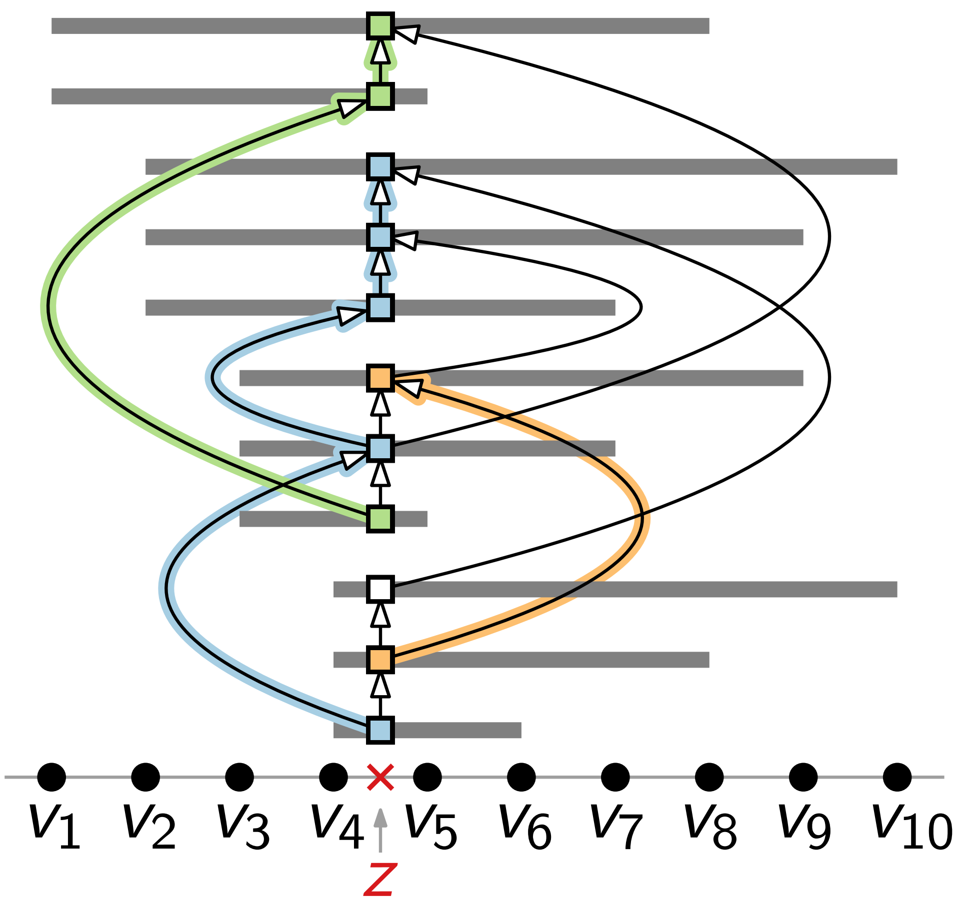


Hitting Set of Size 1

Theorem. Given an ordered graph (G, σ) with n vertices, m edges, and $h(G, \sigma) = 1$, EDGE DELETION TO p-PAGE PLANAR can be solved in $O(m^3 \log n \log \log p)$ time.

Proof.

- Define directed graph.
- Find p directed paths.

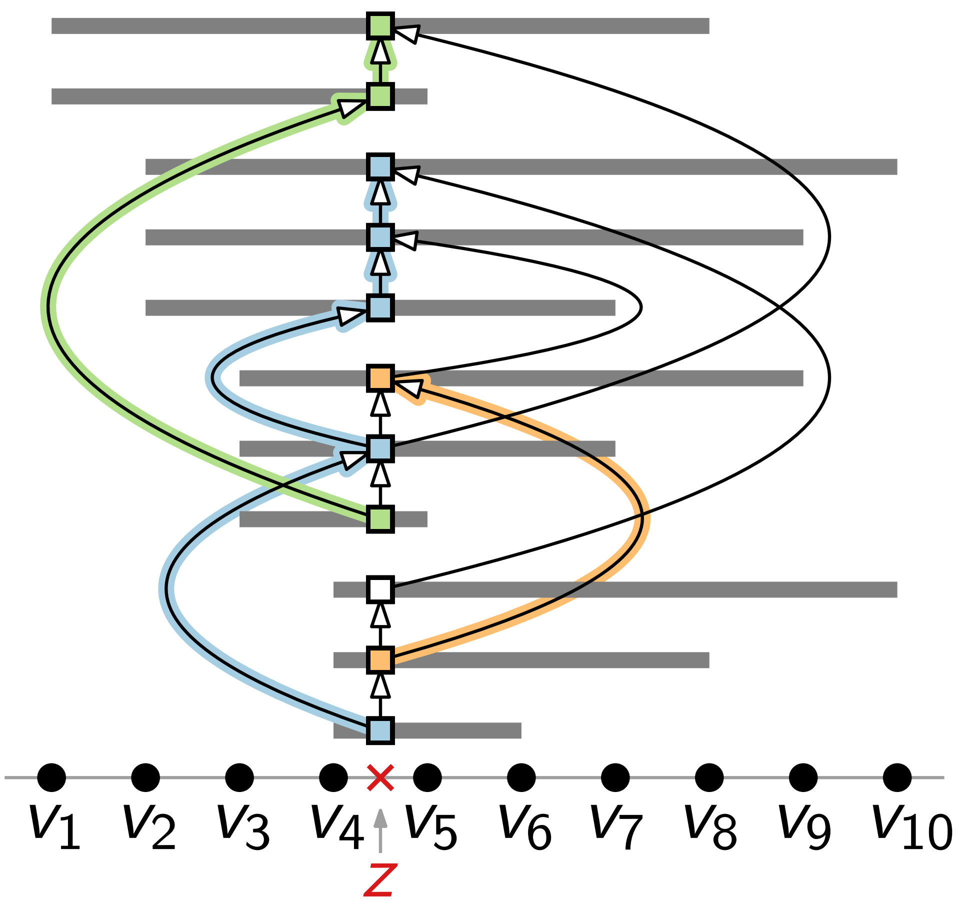


Hitting Set of Size 1

Theorem. Given an ordered graph (G, σ) with n vertices, m edges, and $h(G, \sigma) = 1$, Edge Deletion to p-Page Planar can be solved in $O(m^3 \log n \log\log p)$ time.

Proof.

– Define directed graph.
– Find p directed paths.
– Define flow network.



Hitting Set of Size 1

Theorem. Given an ordered graph (G, σ) with n vertices, m edges, and $h(G, \sigma) = 1$, Edge Deletion to p-Page Planar can be solved in $O(m^3 \log n \log \log p)$ time.

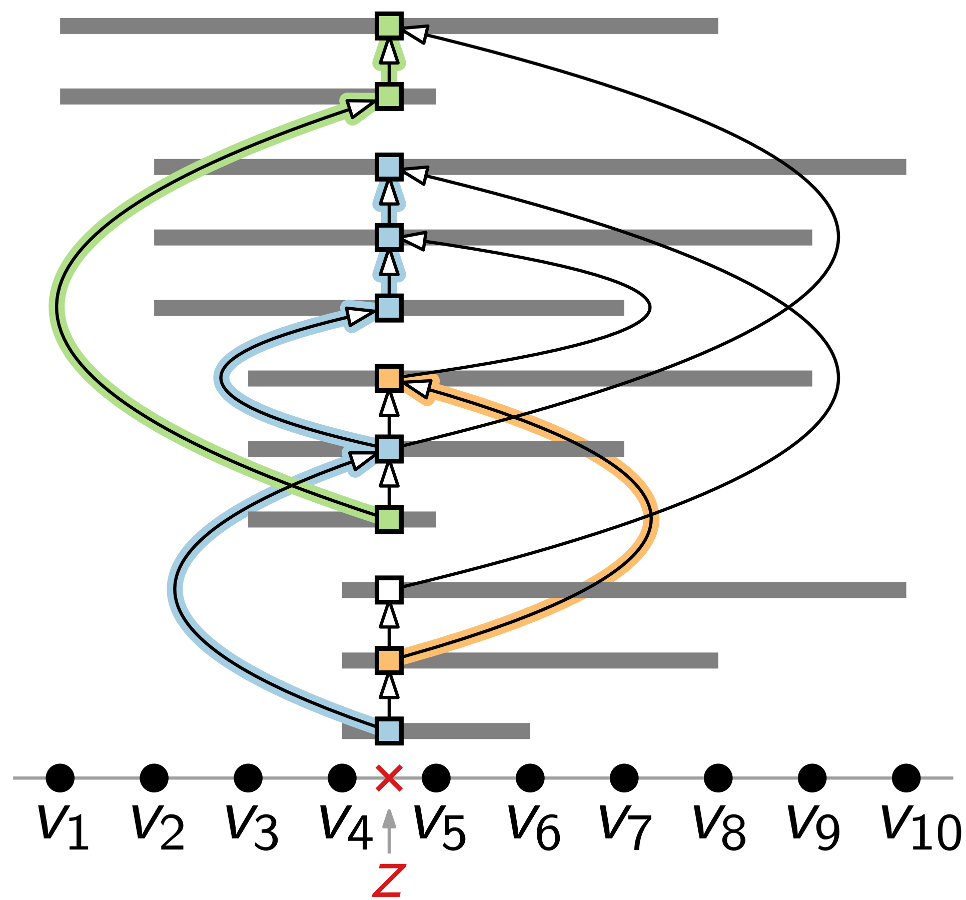
Proof.

– Define directed graph.

– Find p directed paths.

– Define flow network.

– Find min-cost max flow.



Hitting Set of Size 1

Theorem. Given an ordered graph (G, σ) with n vertices, m edges, and $h(G, \sigma) = 1$, EDGE DELETION TO p-PAGE PLANAR can be solved in $O(m^3 \log n \log\log p)$ time.

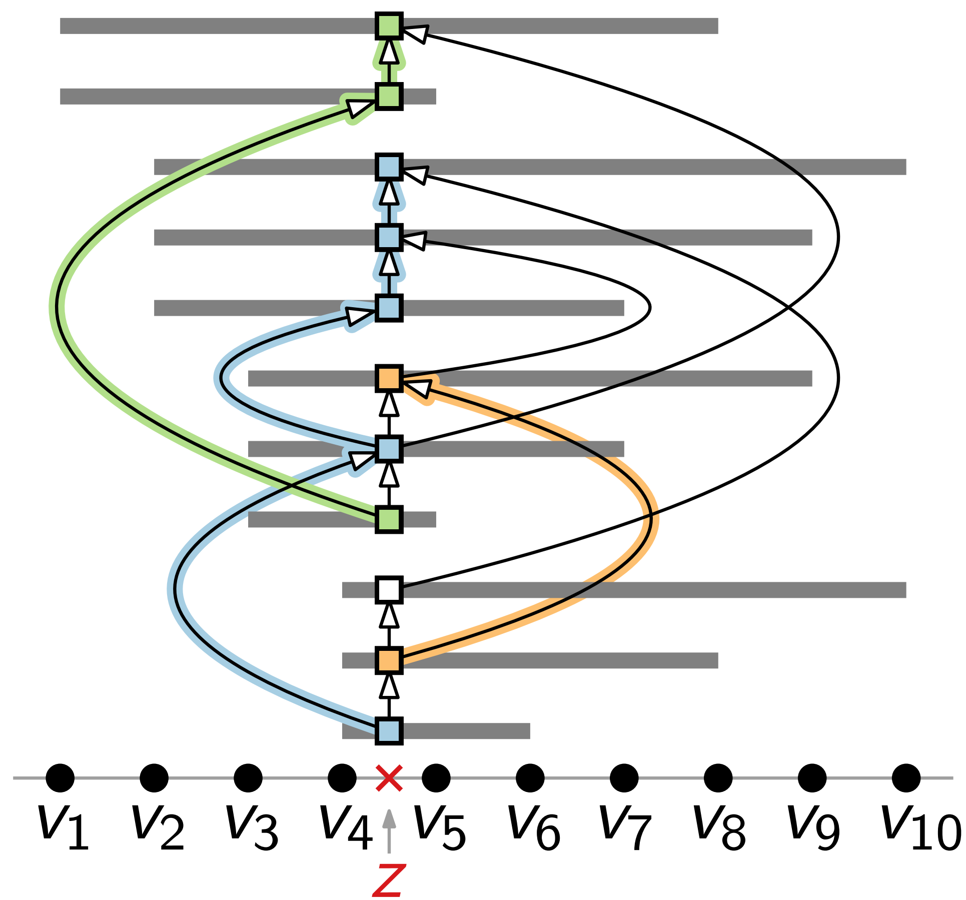
Proof.

– Define directed graph.

– Find p directed paths.

– Define flow network.

– Find min-cost max flow. Such a flow has value p



Hitting Set of Size 1

Theorem. Given an ordered graph (G, σ) with n vertices, m edges, and $h(G, \sigma) = 1$, Edge Deletion to p-Page Planar can be solved in $O(m^3 \log n \log \log p)$ time.

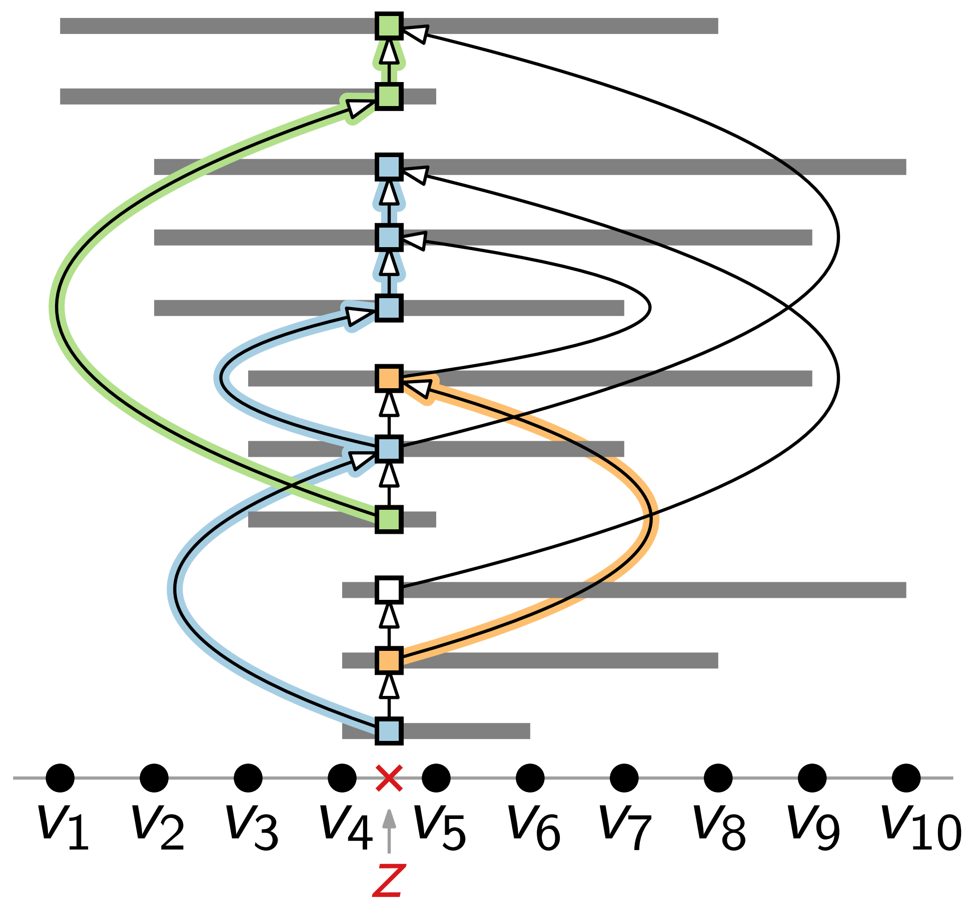
Proof.

– Define directed graph.

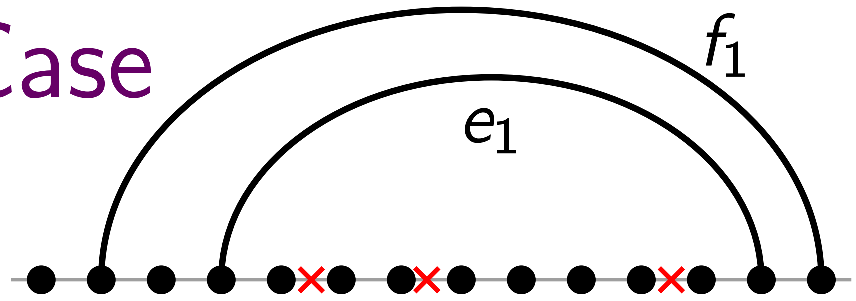
– Find p directed paths.

– Define flow network.

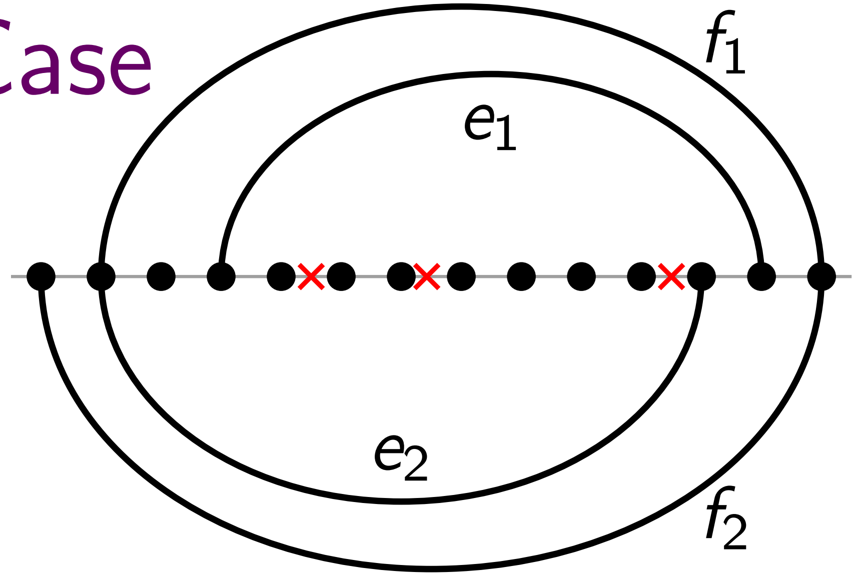
– Find min-cost max flow. Such a flow has value p and max. total path length.



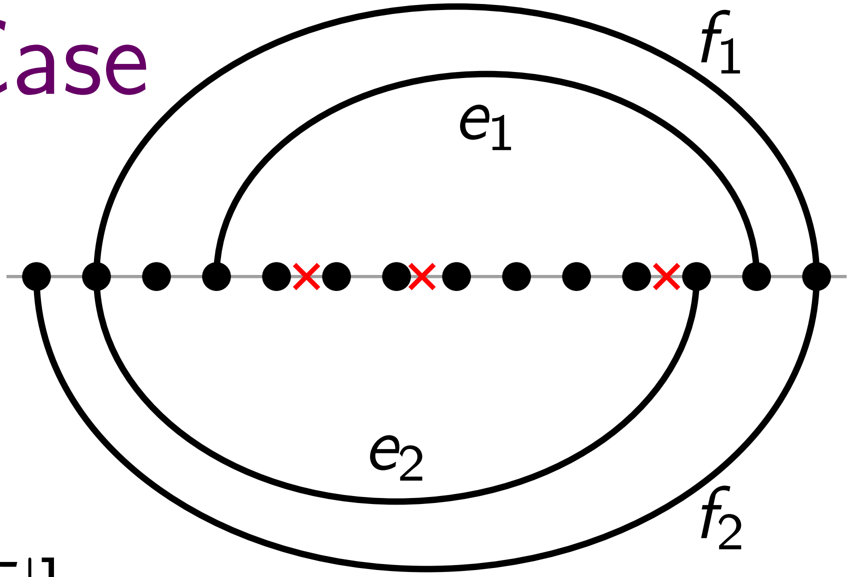
Preparing for the General Case



Preparing for the General Case

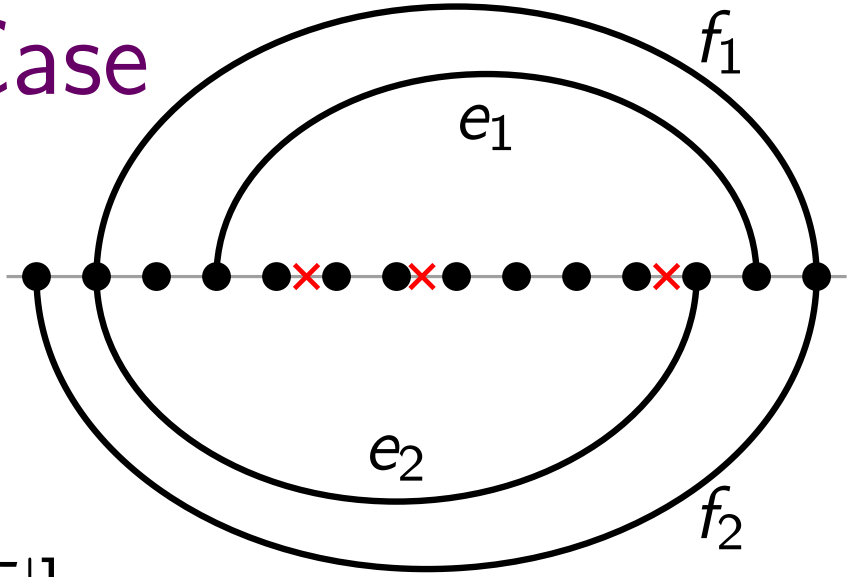


Preparing for the General Case



Two subsets $E, F \subseteq E(G)$ are *compatible* if $|E| = |F|$ and there is an enumeration $e_1, \ldots, e_{|F|}$ of E and an enumeration $f_1, \ldots, f_{|F|}$ of F s.t. e_i is contained in f_i for each $i \in [|F|]$.

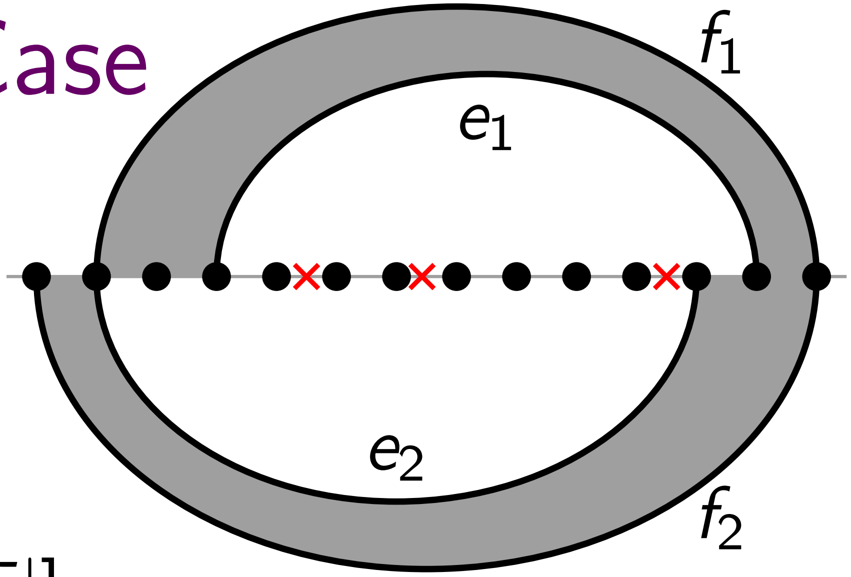
Preparing for the General Case



Two subsets $E, F \subseteq E(G)$ are *compatible* if $|E| = |F|$ and there is an enumeration $e_1, \ldots, e_{|F|}$ of E and an enumeration $f_1, \ldots, f_{|F|}$ of F s.t. e_i is contained in f_i for each $i \in [|F|]$.

Lemma 1. Given an ordered graph (G, σ) with $h(G, \sigma) = 1$ and two subsets $E, F \subseteq E(G)$ of size p, we can decide in $\tilde{O}(m^3)$ time whether E and F are compatible and, if so, solve a version of Edge Deletion to p-Page Planar s.t:

Preparing for the General Case

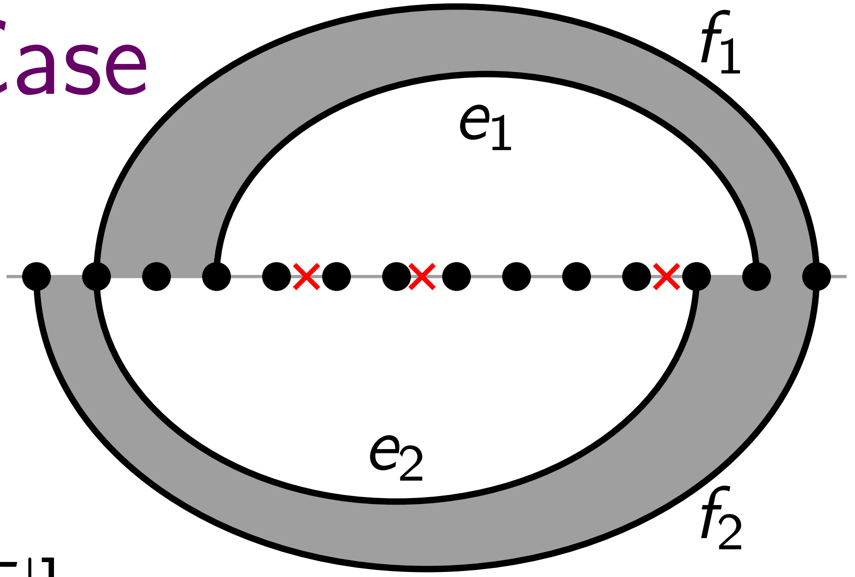


Two subsets $E, F \subseteq E(G)$ are *compatible* if $|E| = |F|$ and there is an enumeration $e_1, \ldots, e_{|F|}$ of E and an enumeration $f_1, \ldots, f_{|F|}$ of F s.t. e_i is contained in f_i for each $i \in [|F|]$.

Lemma 1. Given an ordered graph (G, σ) with $h(G, \sigma) = 1$ and two subsets $E, F \subseteq E(G)$ of size p, we can decide in $\tilde{O}(m^3)$ time whether E and F are compatible and, if so, solve a version of Edge Deletion to p-Page Planar s.t:

- on each page, one edge of E is contained in all other edges and one edge of F contains all other edges on that page.

Preparing for the General Case



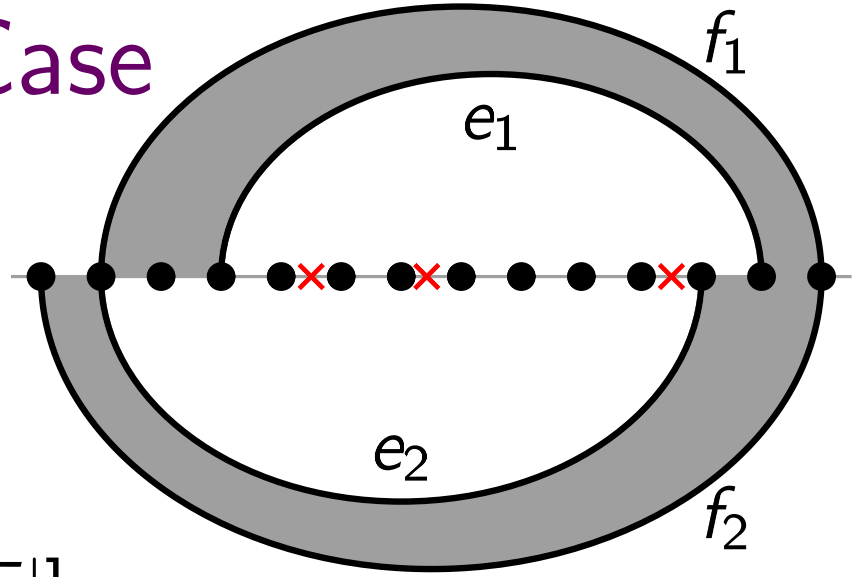
Two subsets $E, F \subseteq E(G)$ are *compatible* if $|E| = |F|$ and there is an enumeration $e_1, \ldots, e_{|F|}$ of E and an enumeration $f_1, \ldots, f_{|F|}$ of F s.t. e_i is contained in f_i for each $i \in [|F|]$.

Lemma 1. Given an ordered graph (G, σ) with $h(G, \sigma) = 1$ and two subsets $E, F \subseteq E(G)$ of size p, we can decide in $\tilde{O}(m^3)$ time whether E and F are compatible and, if so, solve a version of Edge Deletion to p-Page Planar s.t:

- on each page, one edge of E is contained in all other edges and one edge of F contains all other edges on that page.

Proof.

Preparing for the General Case



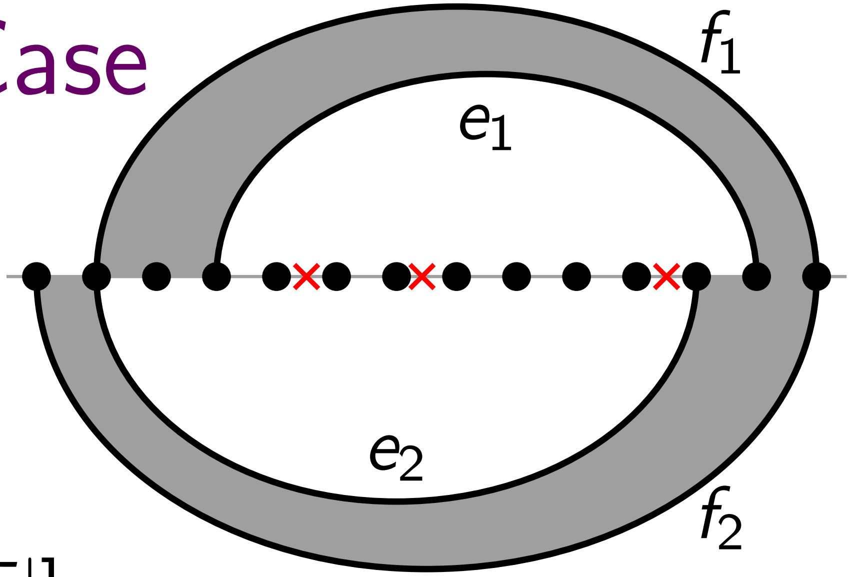
Two subsets $E, F \subseteq E(G)$ are *compatible* if $|E| = |F|$ and there is an enumeration $e_1, \ldots, e_{|F|}$ of E and an enumeration $f_1, \ldots, f_{|F|}$ of F s.t. e_i is contained in f_i for each $i \in [|F|]$.

Lemma 1. Given an ordered graph (G, σ) with $h(G, \sigma) = 1$ and two subsets $E, F \subseteq E(G)$ of size p, we can decide in $\tilde{O}(m^3)$ time whether E and F are compatible and, if so, solve a version of EDGE DELETION TO p-PAGE PLANAR s.t:

- on each page, one edge of E is contained in all other edges and one edge of F contains all other edges on that page.

Proof. Modify flow network:

Preparing for the General Case



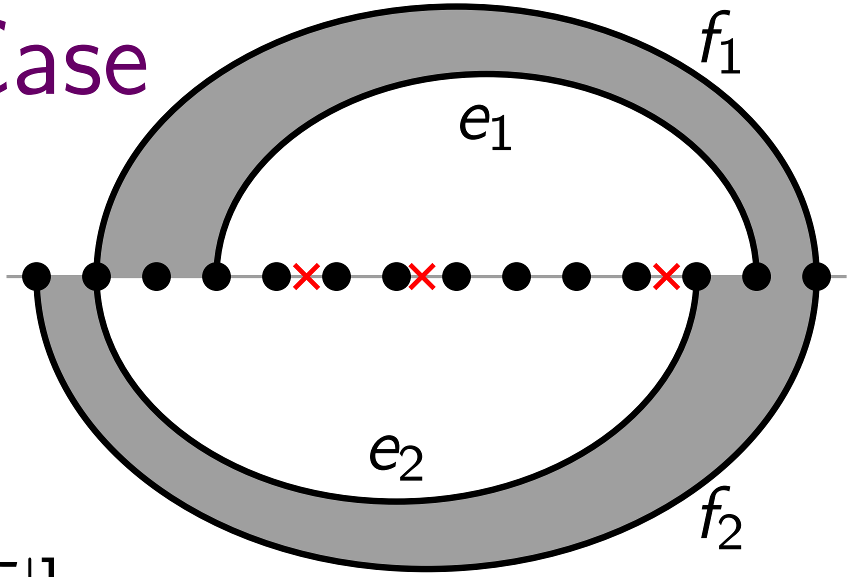
Two subsets $E, F \subseteq E(G)$ are *compatible* if $|E| = |F|$ and there is an enumeration $e_1, \ldots, e_{|F|}$ of E and an enumeration $f_1, \ldots, f_{|F|}$ of F s.t. e_i is contained in f_i for each $i \in [|F|]$.

Lemma 1. Given an ordered graph (G, σ) with $h(G, \sigma) = 1$ and two subsets $E, F \subseteq E(G)$ of size p, we can decide in $\tilde{O}(m^3)$ time whether E and F are compatible and, if so, solve a version of EDGE DELETION TO p-PAGE PLANAR s.t:

- on each page, one edge of E is contained in all other edges and one edge of F contains all other edges on that page.

Proof. Modify flow network: Connect only $s' \to E$ and $F \to t$.

Preparing for the General Case



Two subsets $E, F \subseteq E(G)$ are *compatible* if $|E| = |F|$ and there is an enumeration $e_1, \ldots, e_{|F|}$ of E and an enumeration $f_1, \ldots, f_{|F|}$ of F s.t. e_i is contained in f_i for each $i \in [|F|]$.

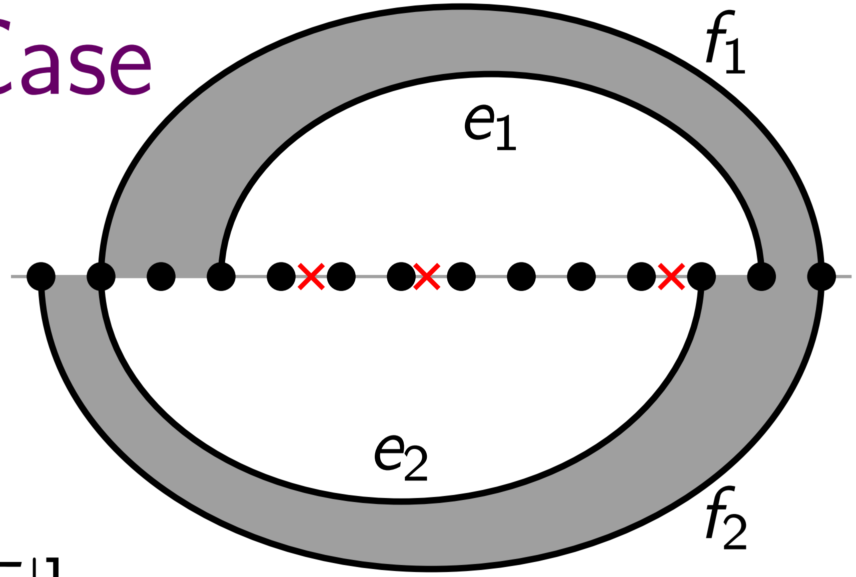
Lemma 1. Given an ordered graph (G, σ) with $h(G, \sigma) = 1$ and two subsets $E, F \subseteq E(G)$ of size p, we can decide in $\tilde{O}(m^3)$ time whether E and F are compatible and, if so, solve a version of Edge Deletion to p-Page Planar s.t:

- on each page, one edge of E is contained in all other edges and one edge of F contains all other edges on that page.

Proof. Modify flow network: Connect only $s' \to E$ and $F \to t$.

E and E compatible $\Leftrightarrow$

Preparing for the General Case



Two subsets $E, F \subseteq E(G)$ are *compatible* if $|E| = |F|$ and there is an enumeration $e_1, \ldots, e_{|F|}$ of E and an enumeration $f_1, \ldots, f_{|F|}$ of F s.t. e_i is contained in f_i for each $i \in [|F|]$.

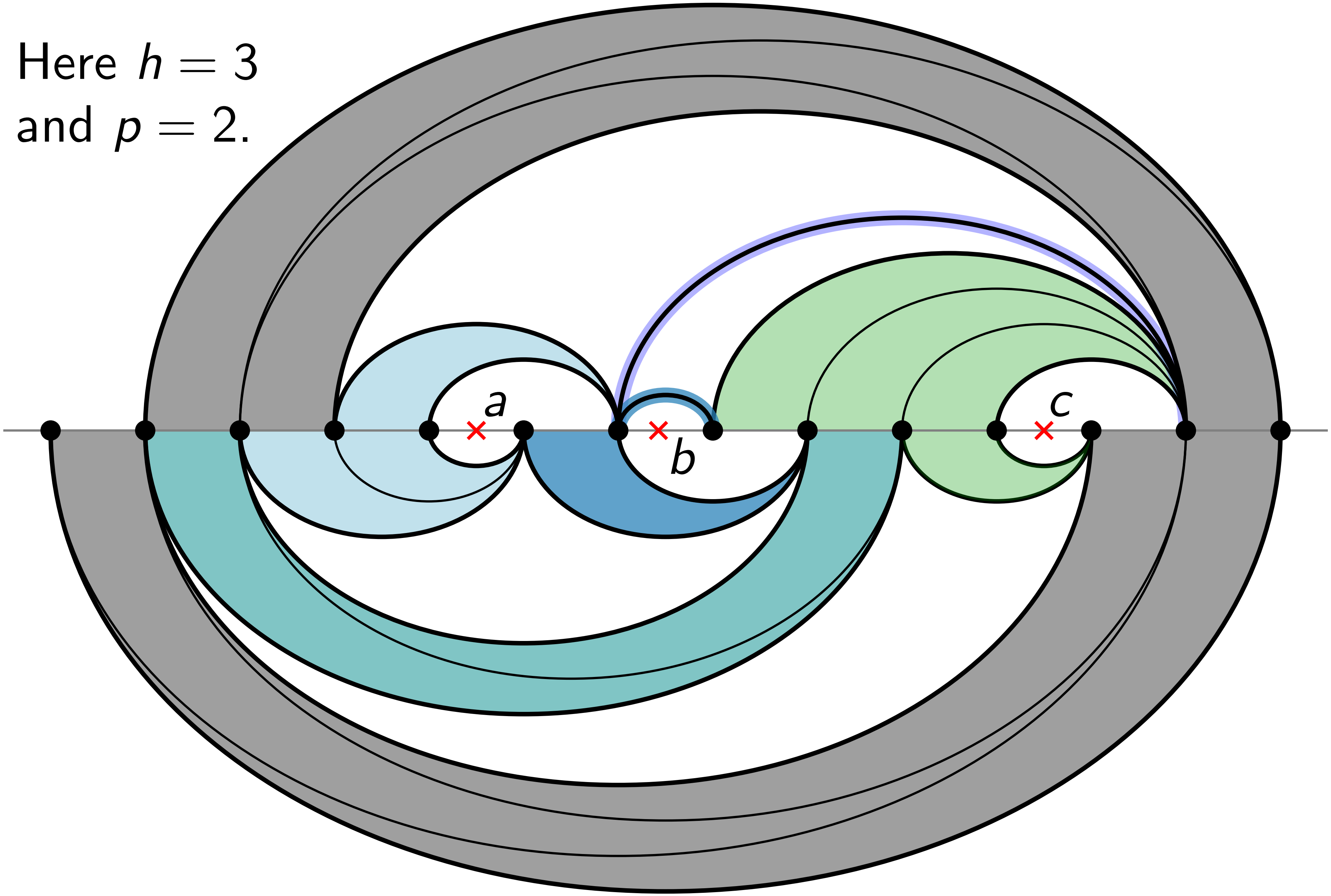
Lemma 1. Given an ordered graph (G, σ) with $h(G, \sigma) = 1$ and two subsets $E, F \subseteq E(G)$ of size p, we can decide in $\tilde{O}(m^3)$ time whether E and F are compatible and, if so, solve a version of Edge Deletion to p-Page Planar s.t:

- on each page, one edge of E is contained in all other edges and one edge of F contains all other edges on that page.

Proof. Modify flow network: Connect only $s' \to E$ and $F \to t$.
E and E compatible $\Leftrightarrow$ maximum flow has value p.

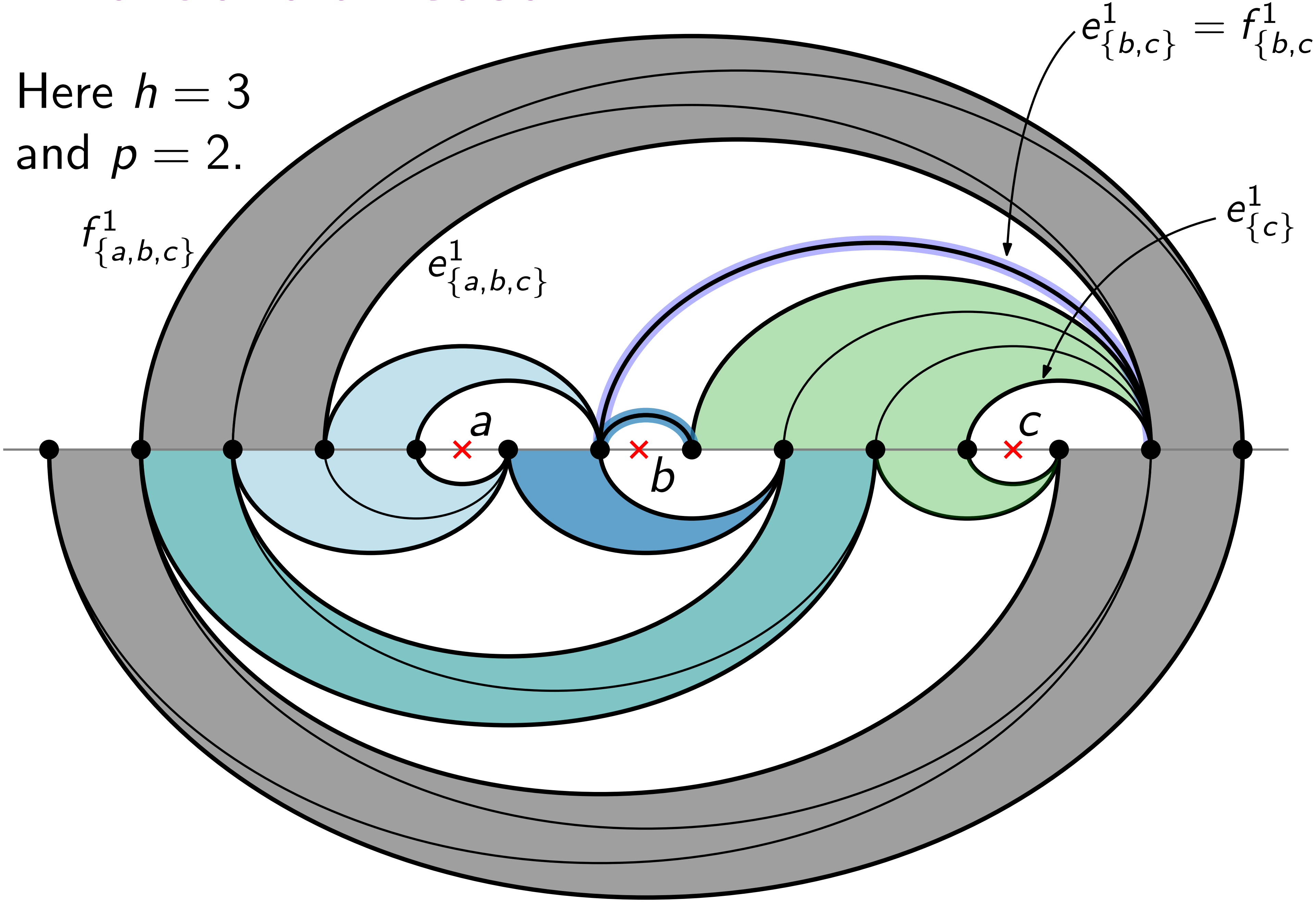
The General Case

Here $h = 3$
and $p = 2$.



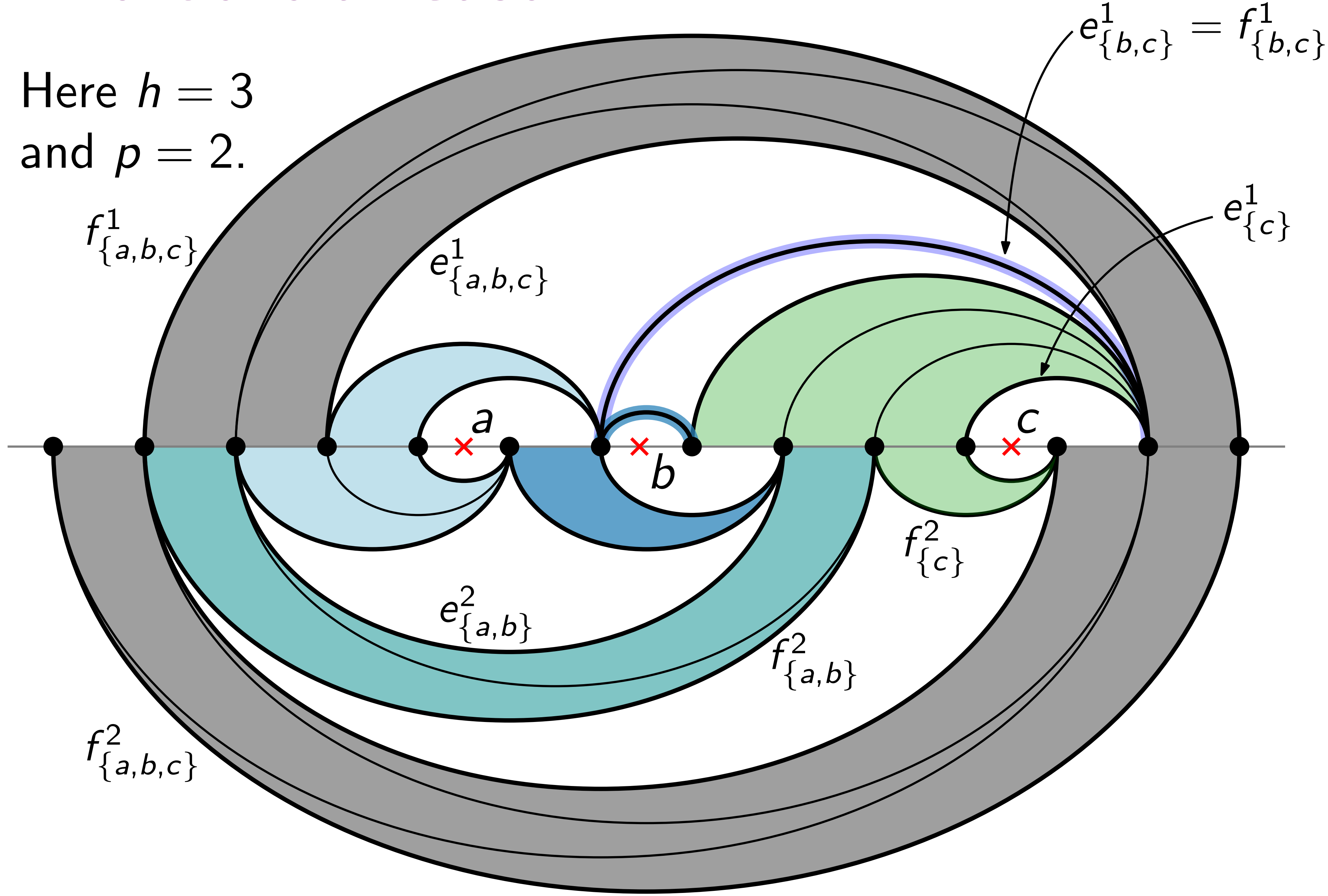
The General Case

Here $h = 3$
and $p = 2$.



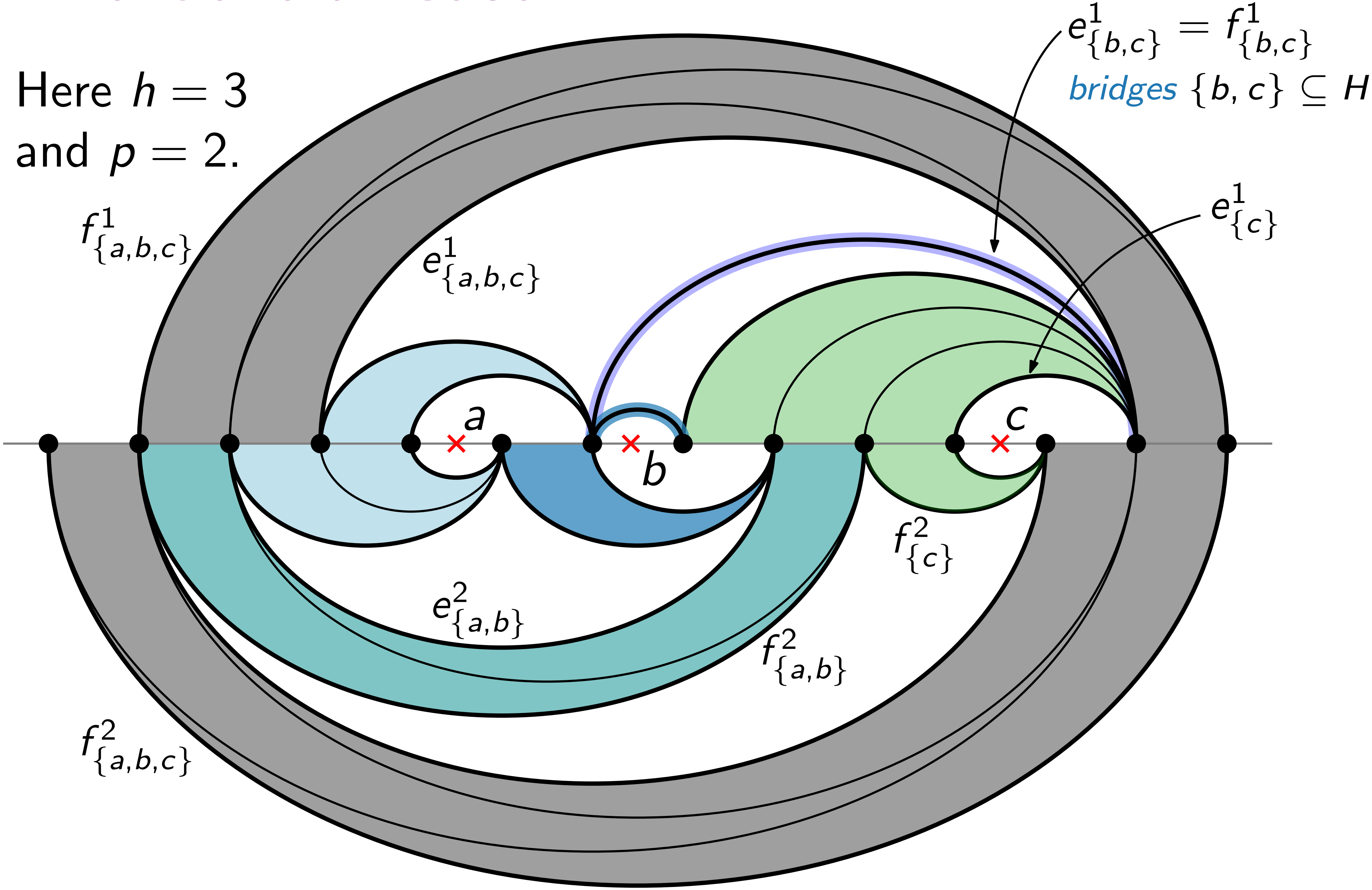
The General Case

Here $h = 3$ and $p = 2$.



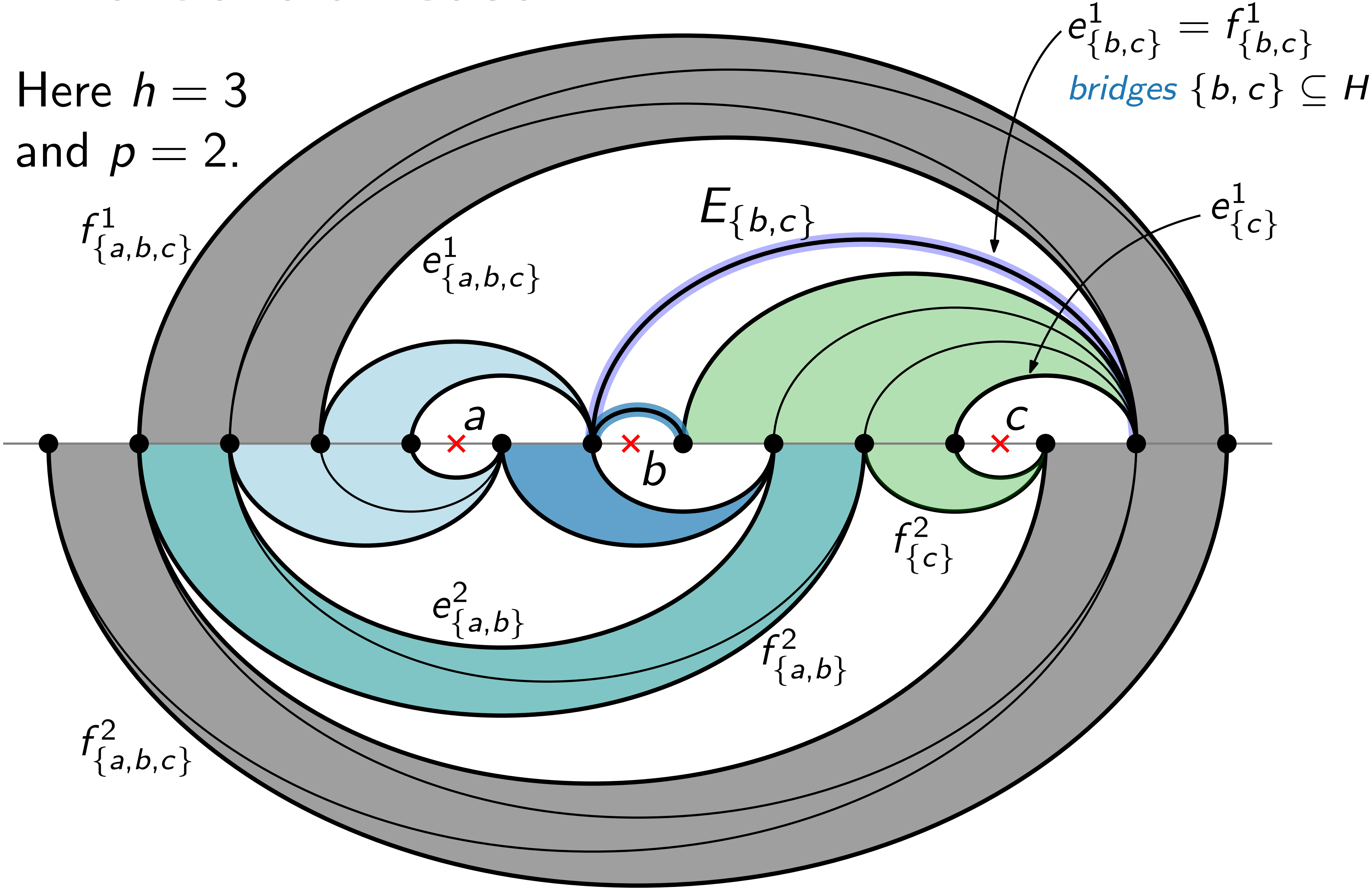
The General Case

Here $h = 3$ and $p = 2$.



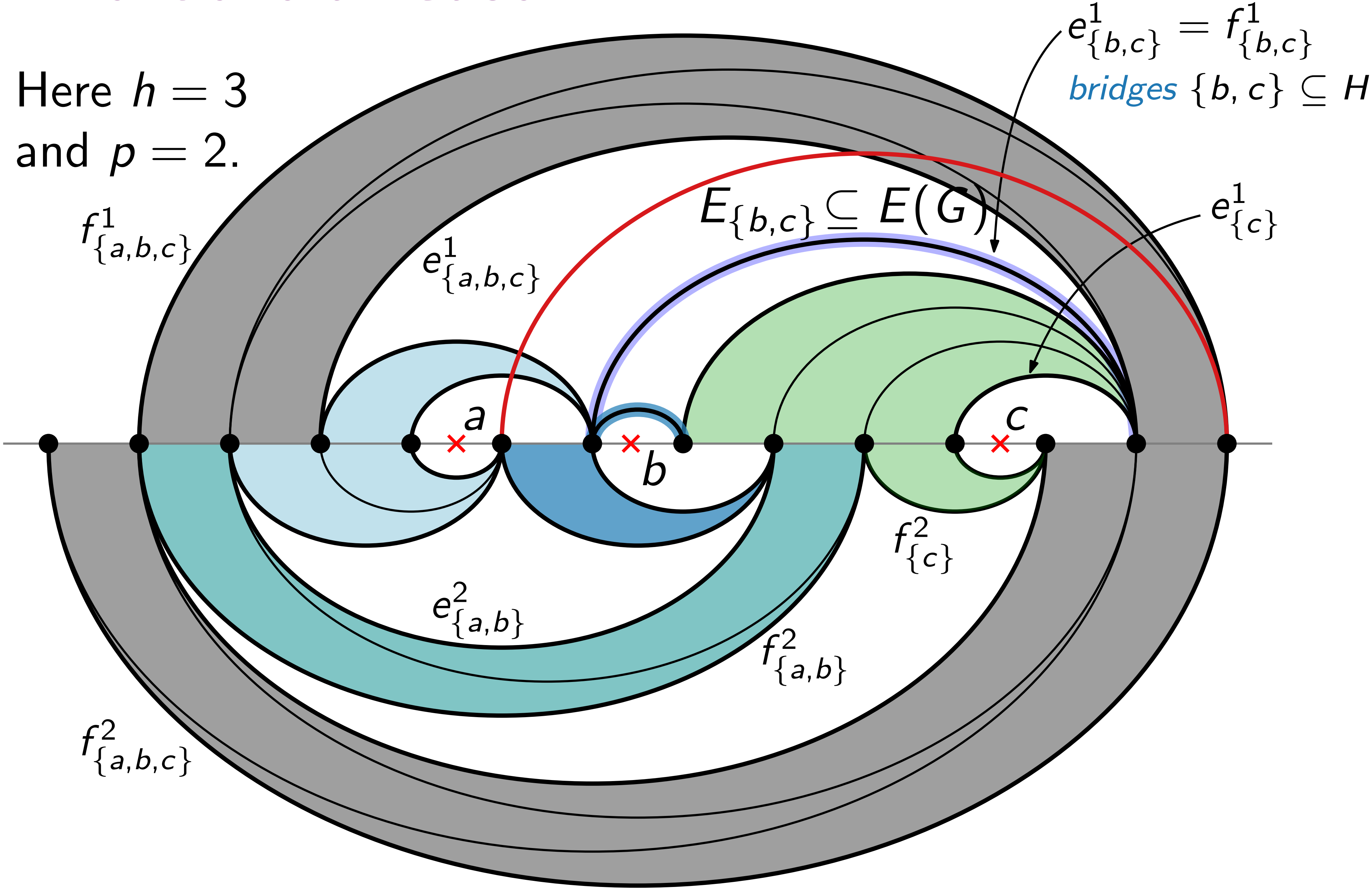
The General Case

Here $h = 3$
and $p = 2$.



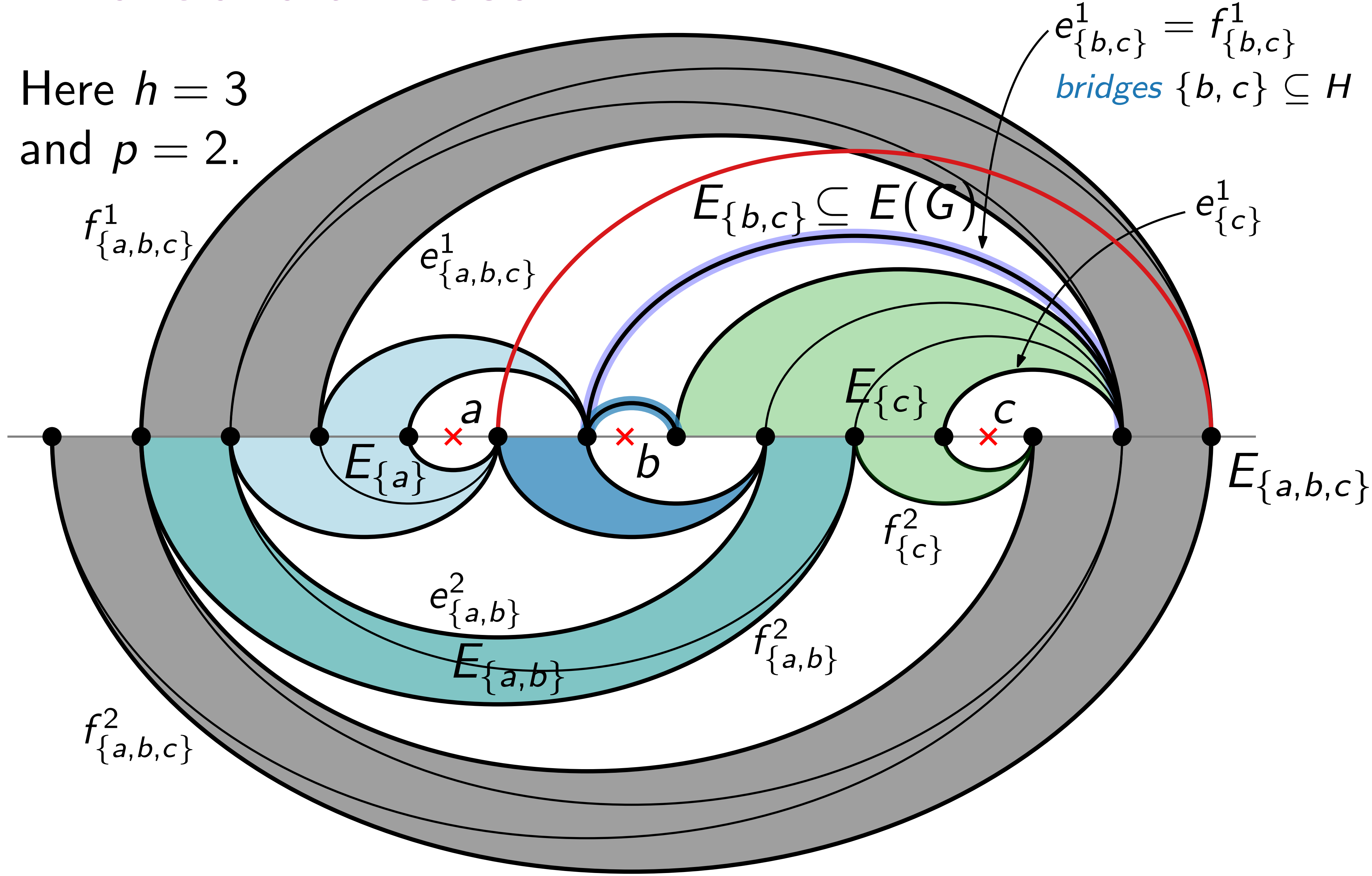
The General Case

Here $h = 3$
and $p = 2$.



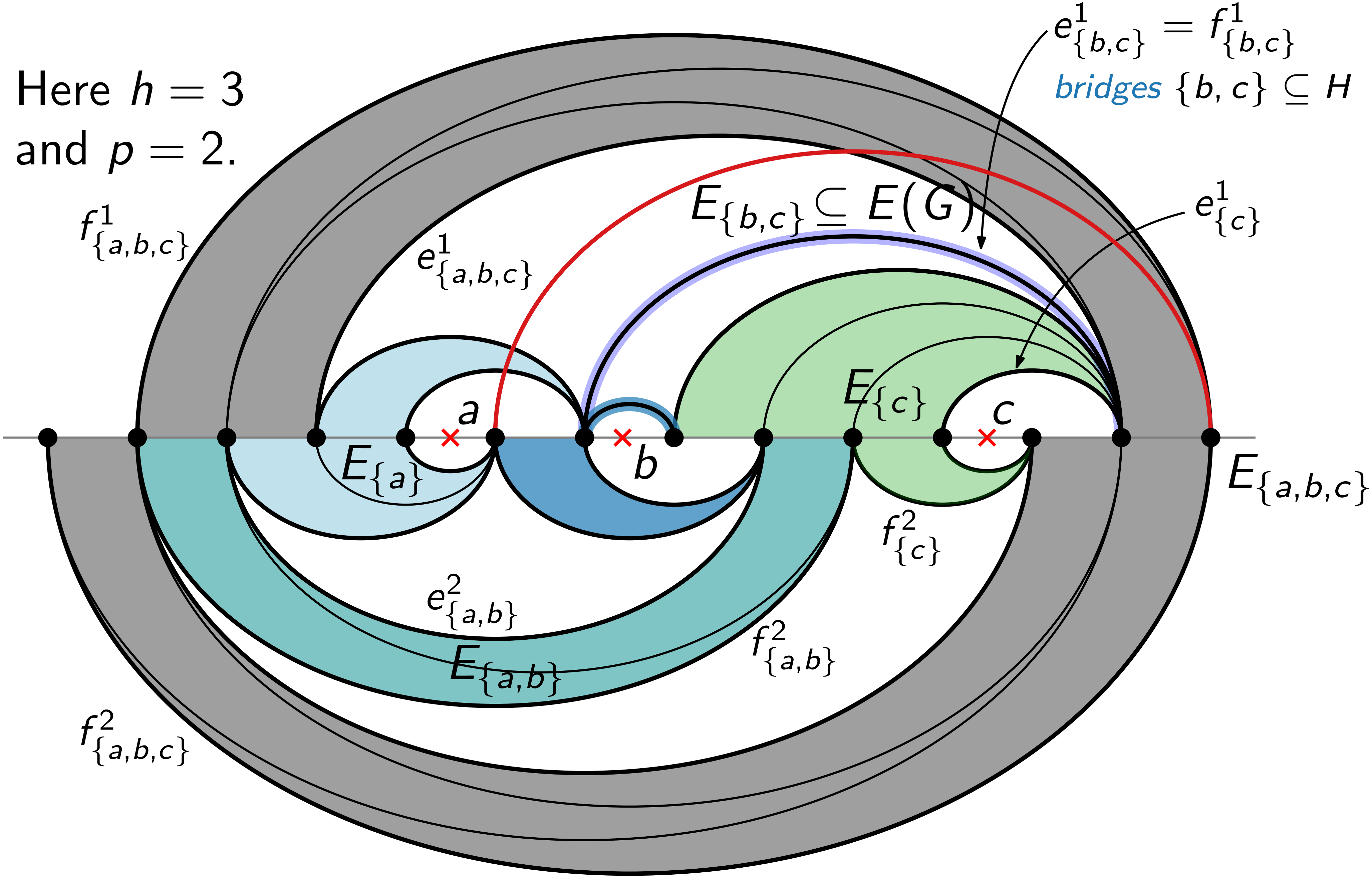
The General Case

Here $h = 3$ and $p = 2$.



The General Case

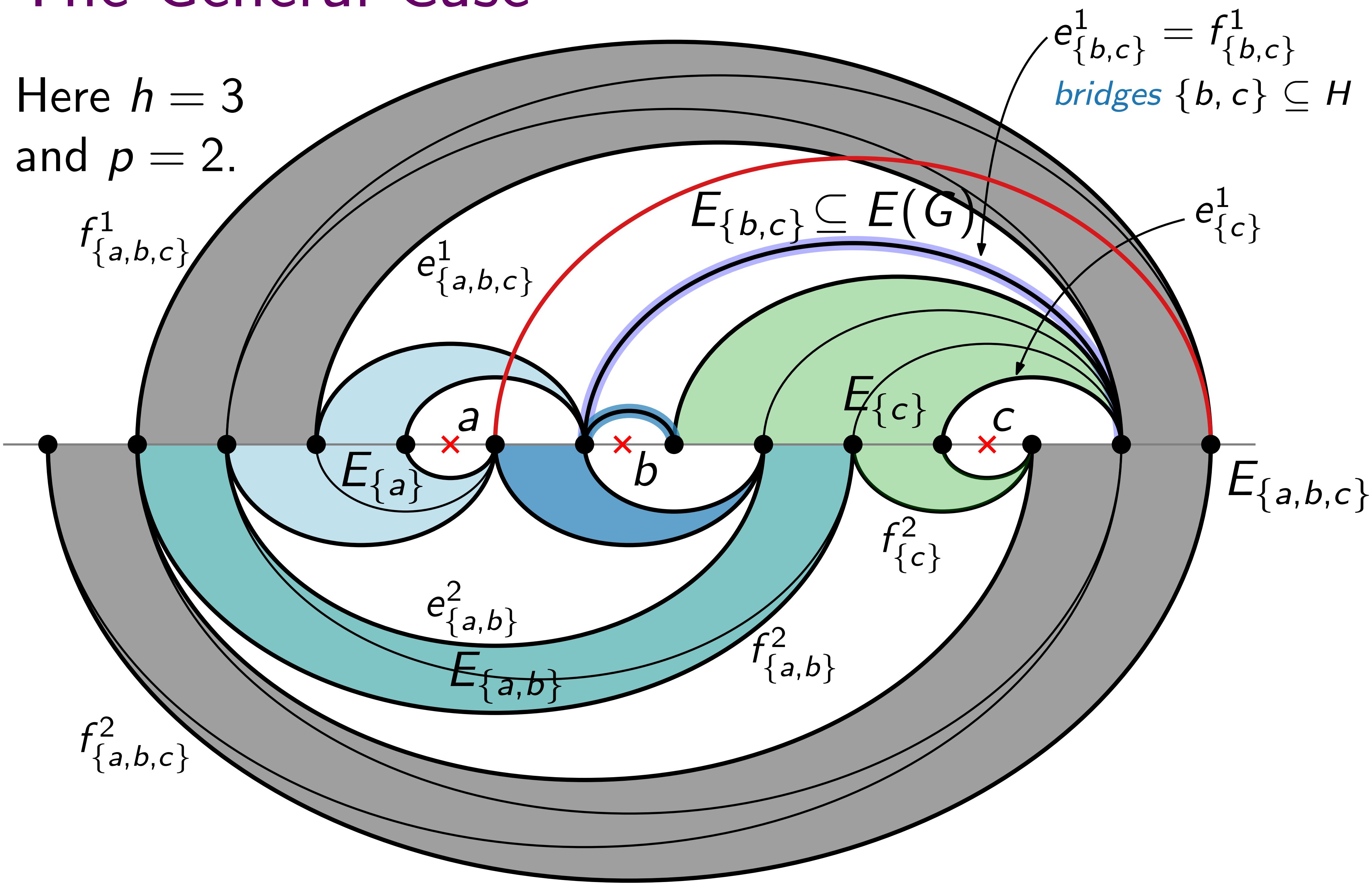
Here $h = 3$
and $p = 2$.



– Split in $(h = 1)$-type instances

The General Case

Here $h = 3$ and $p = 2$.



– Split in $(h = 1)$-type instances – Adjust flow network

Where Is the Difficulty?

For $E \subseteq E(G)$ and $q \in [p]$, let E^q be the edges on page q.

Where Is the Difficulty?

For $E \subseteq E(G)$ and $q \in [p]$, let E^q be the edges on page q.

Let $\mathcal{X}^q$ be the family of subsets of H bridged by edges in E^q.

Where Is the Difficulty?

For $E \subseteq E(G)$ and $q \in [p]$, let E^q be the edges on page q.

Let $\mathcal{X}^q$ be the family of subsets of H bridged by edges in E^q.

If page q is crossing-free, the set family $\mathcal{X}^q$ is *laminar*.

Where Is the Difficulty?

For $E \subseteq E(G)$ and $q \in [p]$, let E^q be the edges on page q.

Let $\mathcal{X}^q$ be the family of subsets of H bridged by edges in E^q.

If page q is crossing-free, the set family $\mathcal{X}^q$ is *laminar*.

$\mathcal{E}^q = \{(X, e_X^q, f_X^q) \mid X \in \mathcal{X}^q\}$ is the *partial encoding* of E on page q

Where Is the Difficulty?

For $E \subseteq E(G)$ and $q \in [p]$, let E^q be the edges on page q.

Let $\mathcal{X}^q$ be the family of subsets of H bridged by edges in E^q.

If page q is crossing-free, the set family $\mathcal{X}^q$ is *laminar*.

$\mathcal{E}^q = \{(X, e_X^q, f_X^q) \mid X \in \mathcal{X}^q\}$ is the *partial encoding* of E on page q and $\langle \mathcal{E}^1, \ldots, \mathcal{E}^p \rangle$ is the *encoding* of E.

Where Is the Difficulty?

For $E \subseteq E(G)$ and $q \in [p]$, let E^q be the edges on page q.

Let $\mathcal{X}^q$ be the family of subsets of H bridged by edges in E^q.

If page q is crossing-free, the set family $\mathcal{X}^q$ is *laminar*.

$\mathcal{E}^q = \{(X, e_X^q, f_X^q) \mid X \in \mathcal{X}^q\}$ is the *partial encoding* of E on page q and $\langle \mathcal{E}^1, \ldots, \mathcal{E}^p \rangle$ is the *encoding* of E.

If $X \subseteq H$ is bridged only on, say, page 1 of an optimal drawing,

Where Is the Difficulty?

For $E \subseteq E(G)$ and $q \in [p]$, let E^q be the edges on page q.

Let $\mathcal{X}^q$ be the family of subsets of H bridged by edges in E^q.

If page q is crossing-free, the set family $\mathcal{X}^q$ is *laminar*.

$\mathcal{E}^q = \{(X, e_X^q, f_X^q) \mid X \in \mathcal{X}^q\}$ is the *partial encoding* of E on page q and $\langle \mathcal{E}^1, \ldots, \mathcal{E}^p \rangle$ is the *encoding* of E.

If $X \subseteq H$ is bridged only on, say, page 1 of an optimal drawing, then we just have to select as many edges as possible (without crossing) from those contained between e_X^1 and f_X^1.

Where Is the Difficulty?

For $E \subseteq E(G)$ and $q \in [p]$, let E^q be the edges on page q.

Let $\mathcal{X}^q$ be the family of subsets of H bridged by edges in E^q.

If page q is crossing-free, the set family $\mathcal{X}^q$ is *laminar*.

$\mathcal{E}^q = \{(X, e^q_X, f^q_X) \mid X \in \mathcal{X}^q\}$ is the *partial encoding* of E on page q and $\langle \mathcal{E}^1, \ldots, \mathcal{E}^p \rangle$ is the *encoding* of E.

If $X \subseteq H$ is bridged only on, say, page 1 of an optimal drawing, then we just have to select as many edges as possible (without crossing) from those contained between e^1_X and f^1_X.

Let $Q_X = \{q \in [p] \colon X \in \mathcal{X}^q\}$.

Where Is the Difficulty?

For $E \subseteq E(G)$ and $q \in [p]$, let E^q be the edges on page q.

Let $\mathcal{X}^q$ be the family of subsets of H bridged by edges in E^q.

If page q is crossing-free, the set family $\mathcal{X}^q$ is *laminar*.

$\mathcal{E}^q = \{(X, e^q_X, f^q_X) \mid X \in \mathcal{X}^q\}$ is the *partial encoding* of E on page q and $\langle \mathcal{E}^1, \ldots, \mathcal{E}^p \rangle$ is the *encoding* of E.

If $X \subseteq H$ is bridged only on, say, page 1 of an optimal drawing, then we just have to select as many edges as possible (without crossing) from those contained between e^1_X and f^1_X.

Let $Q_X = \{q \in [p] \colon X \in \mathcal{X}^q\}$.

Challenge: If $|Q_X| > 1$, the choices of which edges are drawn on which of these pages are not independent.

The Main Lemma

Lemma 2: Let $E \subseteq E(G)$ be a solution with encod. $\langle \mathcal{E}^1, \ldots, \mathcal{E}^p \rangle$.

The Main Lemma

Lemma 2: Let $E \subseteq E(G)$ be a solution with encod. $\langle \mathcal{E}^1, \ldots, \mathcal{E}^p \rangle$.
For every $X \subseteq H$ with $Q_X \neq \emptyset$, let

- $e_X = \{e_X^q \mid q \in Q_X\}$,
- $f_X = \{f_X^q \mid q \in Q_X\}$, and
- $S_X \subseteq E_X$ from applying Lemma 1 w.r.t. e_X, f_X, $p' = |Q_X|$.

The Main Lemma

Lemma 2: Let $E \subseteq E(G)$ be a solution with encod. $\langle \mathcal{E}^1, ..., \mathcal{E}^p \rangle$.
For every $X \subseteq H$ with $Q_X \neq \emptyset$, let
– $e_X = \{e_X^q \mid q \in Q_X\}$,
– $f_X = \{f_X^q \mid q \in Q_X\}$, and
– $S_X \subseteq E_X$ from applying Lemma 1 w.r.t. e_X, f_X, $p' = |Q_X|$.
Then $S = \bigcup_X S_X$ is a solution for p pages and $|S| \geq |E|$.

The Main Lemma

Lemma 2: Let $E \subseteq E(G)$ be a solution with encod. $\langle \mathcal{E}^1, ..., \mathcal{E}^p \rangle$.
For every $X \subseteq H$ with $Q_X \neq \emptyset$, let
- $e_X = \{e_X^q \mid q \in Q_X\}$,
- $f_X = \{f_X^q \mid q \in Q_X\}$, and
- $S_X \subseteq E_X$ from applying Lemma 1 w.r.t. e_X, f_X, $p' = |Q_X|$.

Then $S = \bigcup_X S_X$ is a solution for p pages and $|S| \geq |E|$.

Proof. Let $X \subseteq H$ with $Q_X \neq \emptyset$.

The Main Lemma

Lemma 2: Let $E \subseteq E(G)$ be a solution with encod. $\langle \mathcal{E}^1, ..., \mathcal{E}^p \rangle$.
For every $X \subseteq H$ with $Q_X \neq \emptyset$, let
- $e_X = \{e_X^q \mid q \in Q_X\}$,
- $f_X = \{f_X^q \mid q \in Q_X\}$, and
- $S_X \subseteq E_X$ from applying Lemma 1 w.r.t. e_X, f_X, $p' = |Q_X|$.

Then $S = \bigcup_X S_X$ is a solution for p pages and $|S| \geq |E|$.

Proof. Let $X \subseteq H$ with $Q_X \neq \emptyset$.
For $q \in Q_X$, let S_X^q be the edges in S_X that appear on the same page as $e_X^q \in e_X$ when applying Lemma 1.

The Main Lemma

Lemma 2: Let $E \subseteq E(G)$ be a solution with encod. $\langle \mathcal{E}^1, ..., \mathcal{E}^p \rangle$.

For every $X \subseteq H$ with $Q_X \neq \emptyset$, let

– $e_X = \{e_X^q \mid q \in Q_X\}$,

– $f_X = \{f_X^q \mid q \in Q_X\}$, and

– $S_X \subseteq E_X$ from applying Lemma 1 w.r.t. e_X, f_X, $p' = |Q_X|$.

Then $S = \bigcup_X S_X$ is a solution for p pages and $|S| \geq |E|$.

Proof. Let $X \subseteq H$ with $Q_X \neq \emptyset$.

For $q \in Q_X$, let S_X^q be the edges in S_X that appear on the same page as $e_X^q \in e_X$ when applying Lemma 1.

Let $\sigma \colon Q_X \to Q_X$ be the permutation s.t. $f_X^{\sigma(q)}$ is the unique element of f_X in S_X^q.

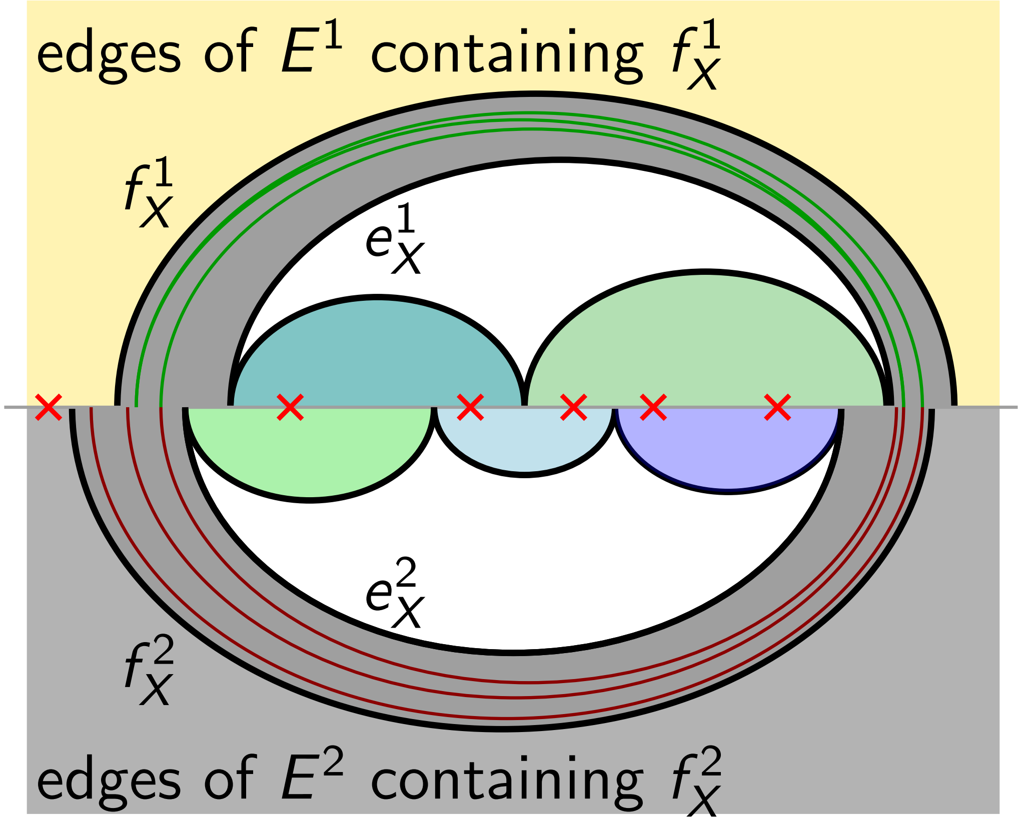
The Main Lemma

Lemma 2: Let $E \subseteq E(G)$ be a solution with encod. $\langle \mathcal{E}^1, \ldots, \mathcal{E}^p \rangle$.
For every $X \subseteq H$ with $Q_X \neq \emptyset$, let
- $e_X = \{e_X^q \mid q \in Q_X\}$,
- $f_X = \{f_X^q \mid q \in Q_X\}$, and
- $S_X \subseteq E_X$ from applying Lemma 1 w.r.t. e_X, f_X, $p' = |Q_X|$.

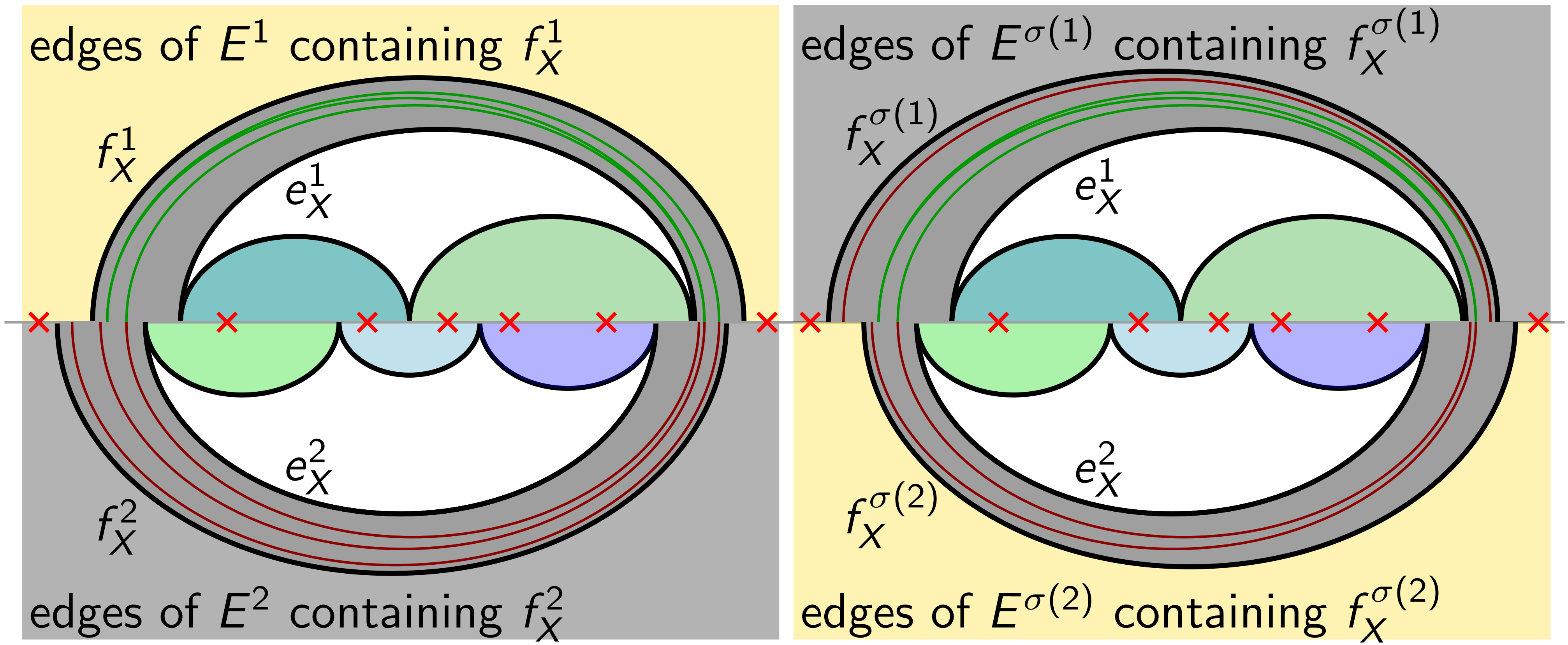
Then $S = \bigcup_X S_X$ is a solution for p pages and $|S| \geq |E|$.

Proof. Let $X \subseteq H$ with $Q_X \neq \emptyset$.
For $q \in Q_X$, let S_X^q be the edges in S_X that appear on the same page as $e_X^q \in e_X$ when applying Lemma 1.
Let $\sigma\colon Q_X \to Q_X$ be the permutation s.t. $f_X^{\sigma(q)}$ is the unique element of f_X in S_X^q.
We make a drawing of $\hat{E} := (E \setminus E_X) \cup S_X$ on p pages by assigning edges to pages, as follows.

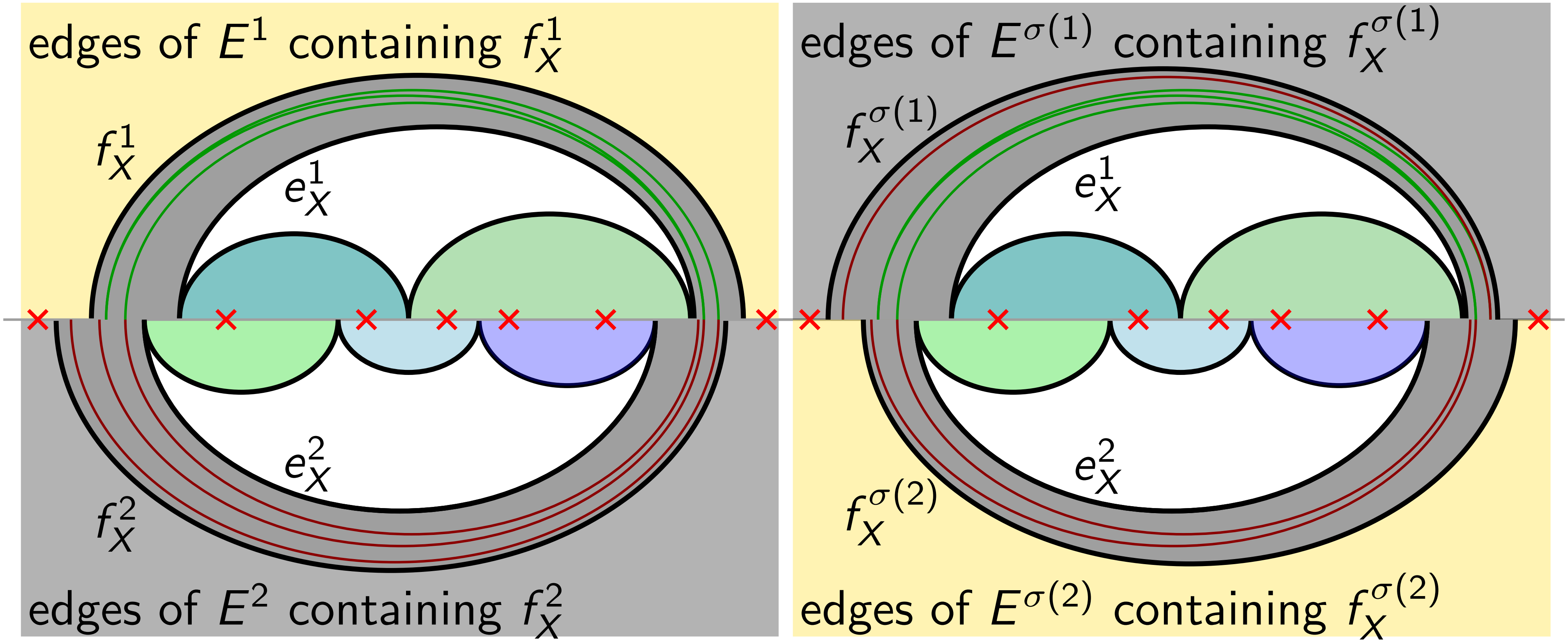
Converting Solution E via $\hat{E}$ into S



Converting Solution E via $\hat{E}$ into S

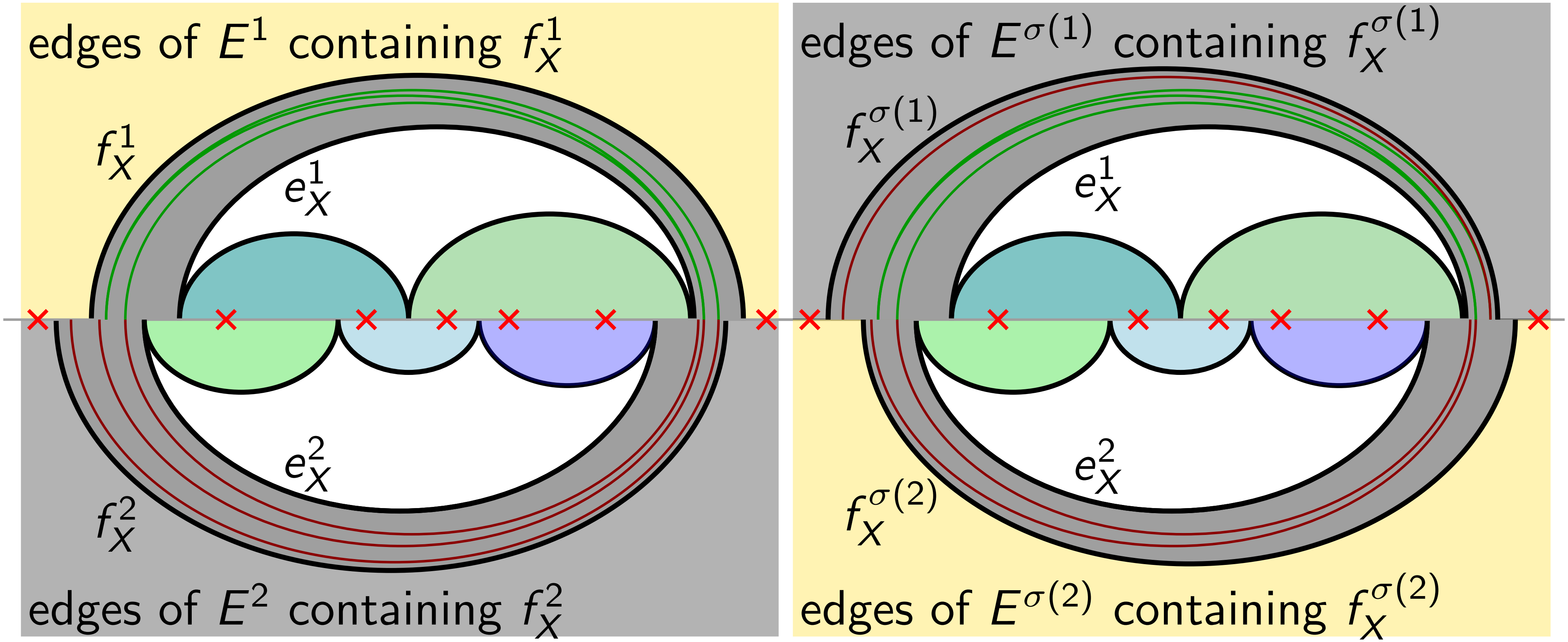


Converting Solution E via $\hat{E}$ into S



For $q \in [p] \setminus Q_X$, set $\hat{E}^q = E^q$.

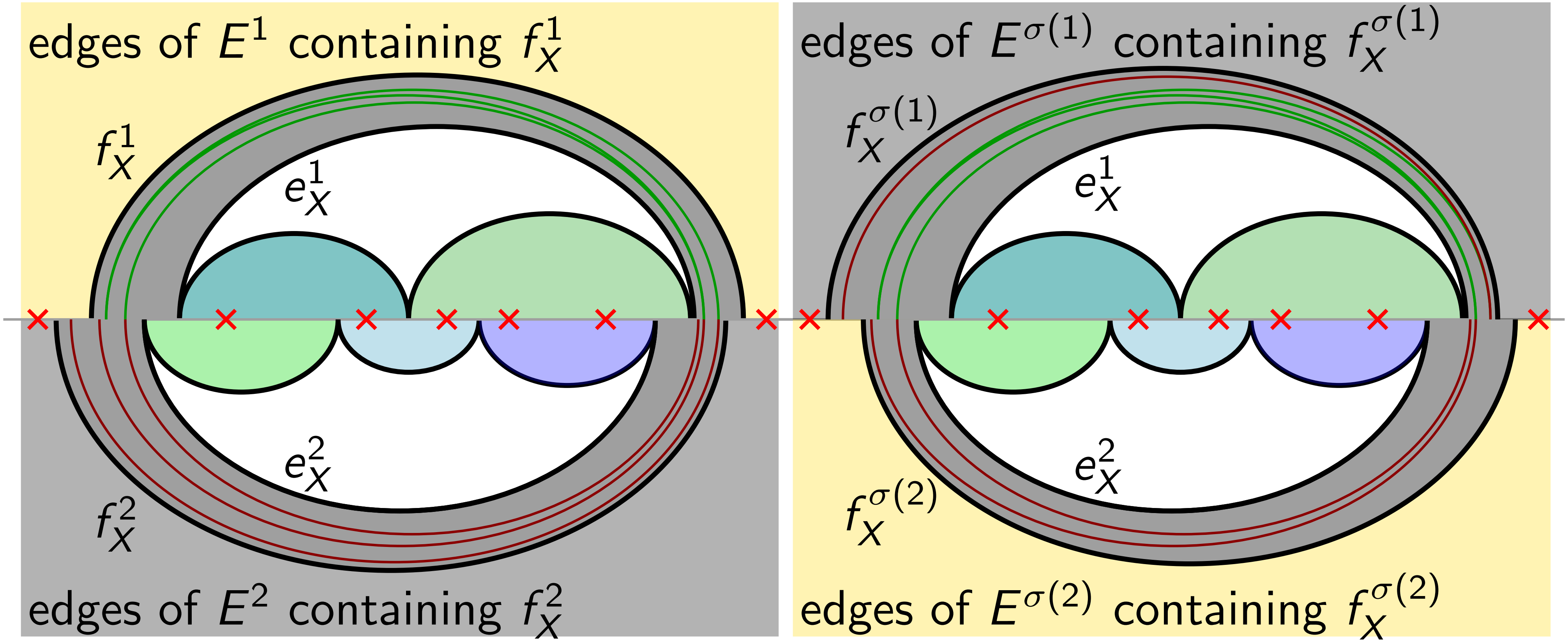
Converting Solution E via $\hat{E}$ into S



For $q \in [p] \setminus Q_X$, set $\hat{E}^q = E^q$.

For $q \in Q_X$, construct $\hat{E}^q$ from $E^{\sigma(q)}$:
- remove the edges contained in $f_X^{\sigma(q)}$,
- add the edges of S_X^q, and
- add the edges of E^q contained in e_X^q.

Converting Solution E via $\hat{E}$ into S



For $q \in [p] \setminus Q_X$, set $\hat{E}^q = E^q$.

For $q \in Q_X$, construct $\hat{E}^q$ from $E^{\sigma(q)}$:

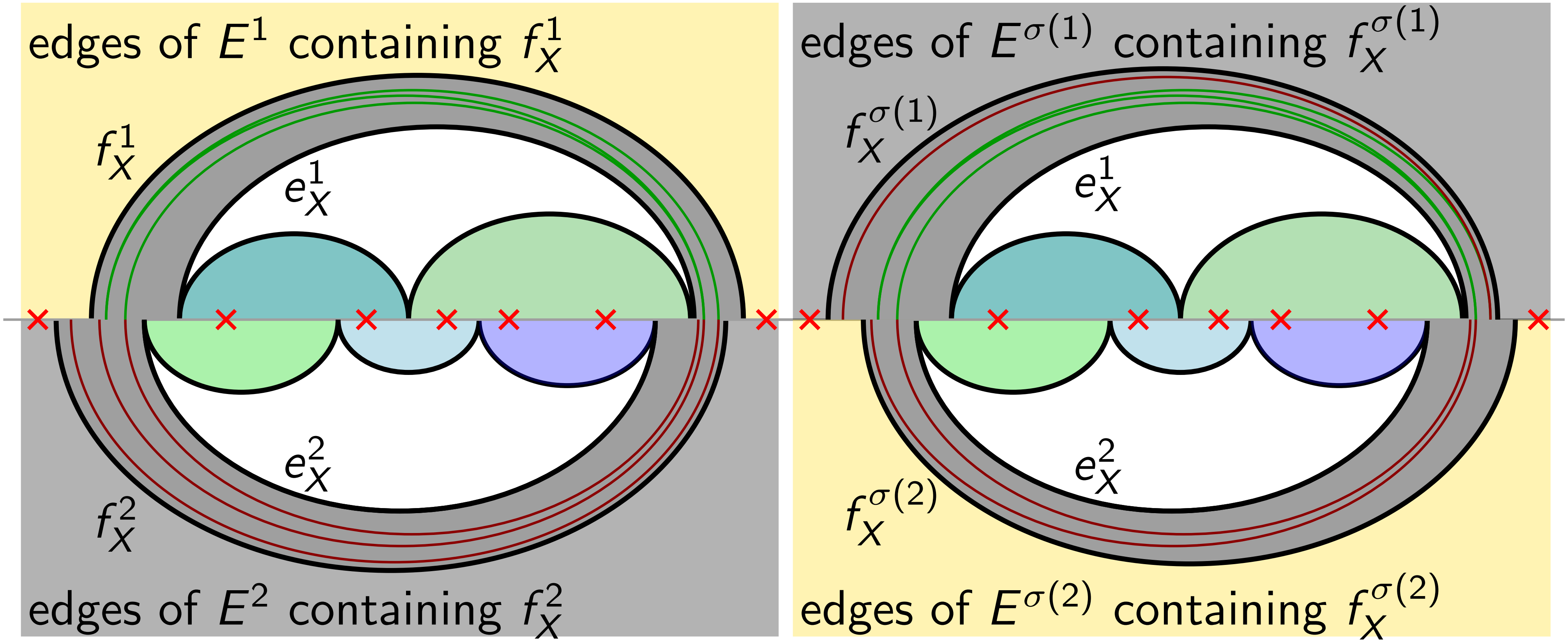
– remove the edges contained in $f_X^{\sigma(q)}$,

– add the edges of S_X^q, and

– add the edges of E^q contained in e_X^q.

$E \cap E_x$ is a feasible sol. of Lem. 1.

Converting Solution E via $\hat{E}$ into S

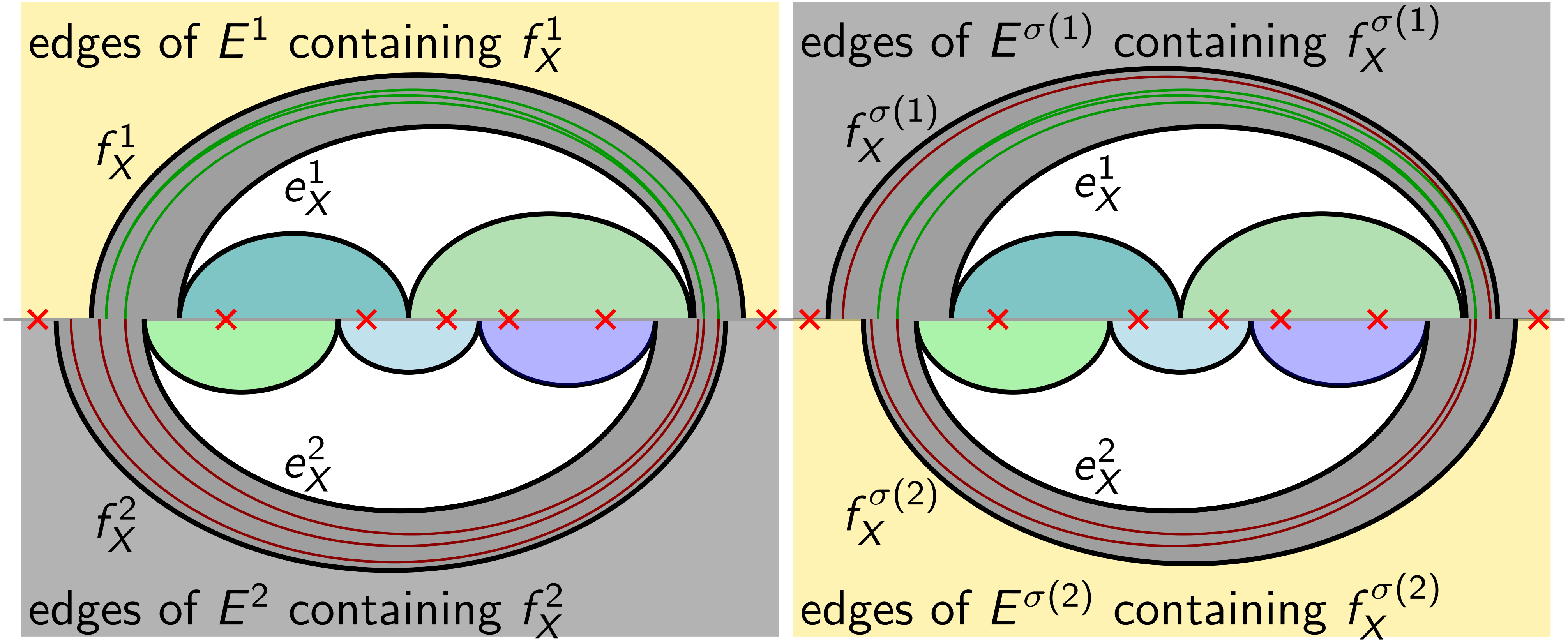


For $q \in [p] \setminus Q_X$, set $\hat{E}^q = E^q$.

For $q \in Q_X$, construct $\hat{E}^q$ from $E^{\sigma(q)}$:
- remove the edges contained in $f_X^{\sigma(q)}$,
- add the edges of S_X^q, and
- add the edges of E^q contained in e_X^q.

$E \cap E_x$ is a feasible sol. of Lem. 1.
$\Rightarrow |E \cap E_x| \leq |S_X|$.

Converting Solution E via $\hat{E}$ into S



For $q \in [p] \setminus Q_X$, set $\hat{E}^q = E^q$.

For $q \in Q_X$, construct $\hat{E}^q$ from $E^{\sigma(q)}$:

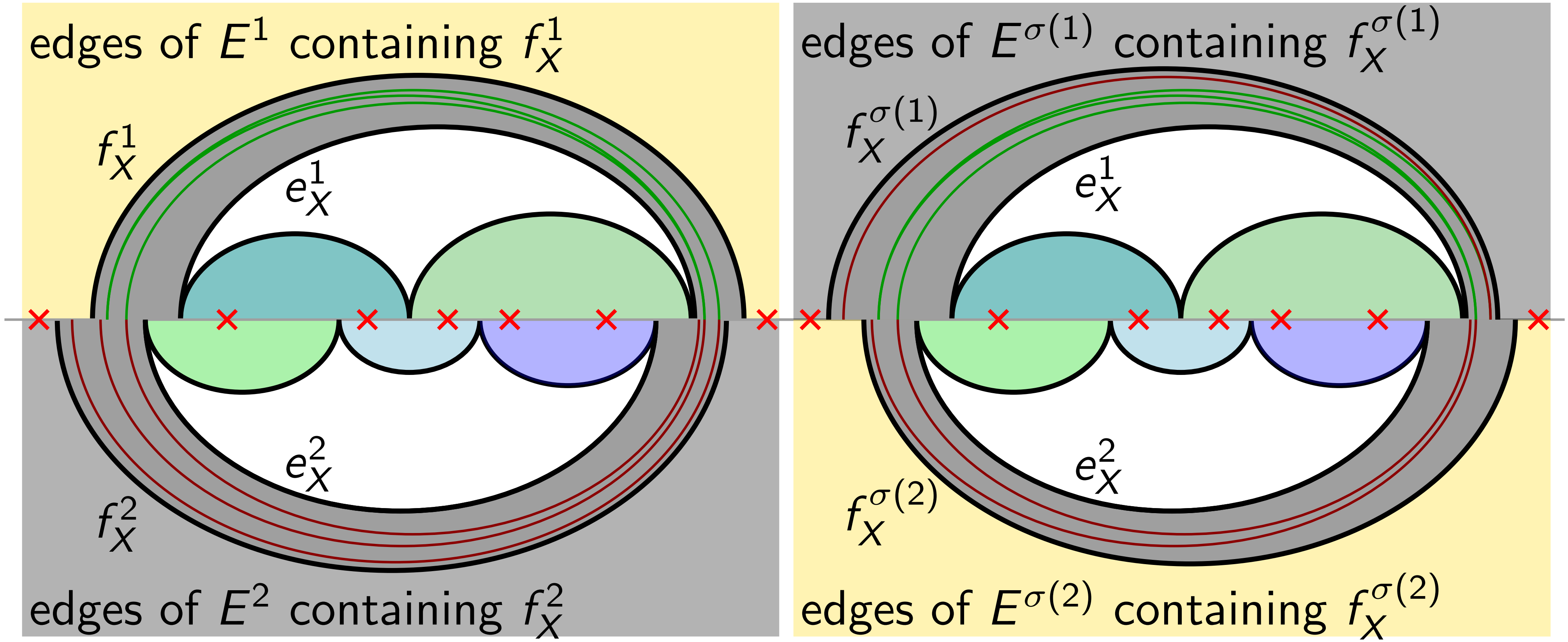
– remove the edges contained in $f_X^{\sigma(q)}$,
– add the edges of S_X^q, and
– add the edges of E^q contained in e_X^q.

$E \cap E_x$ is a feasible sol. of Lem. 1.

$\Rightarrow |E \cap E_x| \leq |S_X|$.

$\Rightarrow |E| \leq |\hat{E}| = |(E \setminus E_X) \cup S_X|$.

Converting Solution E via $\hat{E}$ into S

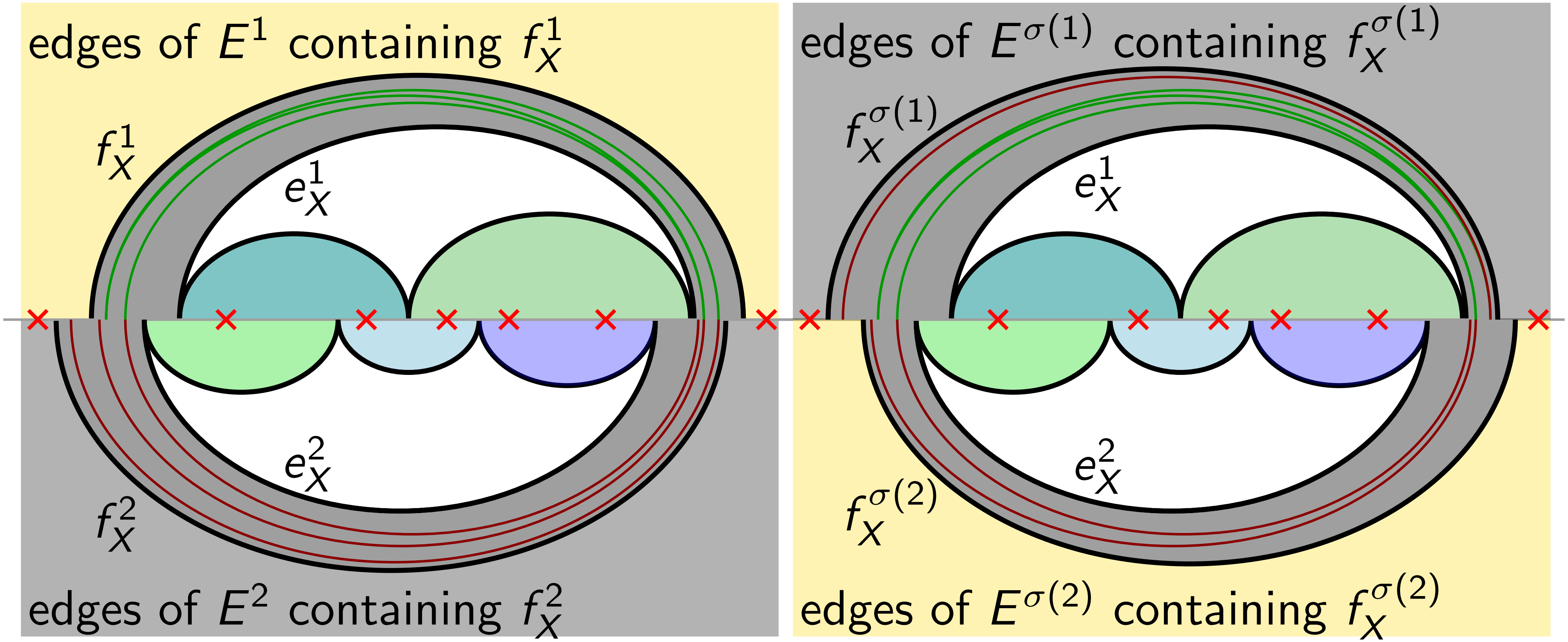


For $q \in [p] \setminus Q_X$, set $\hat{E}^q = E^q$.

For $q \in Q_X$, construct $\hat{E}^q$ from $E^{\sigma(q)}$:
- remove the edges contained in $f_X^{\sigma(q)}$,
- add the edges of S_X^q, and
- add the edges of E^q contained in e_X^q.

$E \cap E_x$ is a feasible sol. of Lem. 1.
$\Rightarrow |E \cap E_x| \leq |S_X|$.
$\Rightarrow |E| \leq |\hat{E}| = |(E \setminus E_X) \cup S_X|$.
Iterate this for each $X \subseteq H$.

Converting Solution E via $\hat{E}$ into S



For $q \in [p] \setminus Q_X$, set $\hat{E}^q = E^q$.

For $q \in Q_X$, construct $\hat{E}^q$ from $E^{\sigma(q)}$:

– remove the edges contained in $f_X^{\sigma(q)}$,

– add the edges of S_X^q, and

– add the edges of E^q contained in e_X^q.

$E \cap E_x$ is a feasible sol. of Lem. 1.

$\Rightarrow |E \cap E_x| \leq |S_X|$.

$\Rightarrow |E| \leq |\hat{E}| = |(E \setminus E_X) \cup S_X|$.

Iterate this for each $X \subseteq H$.

Finally, $\hat{E} = \bigcup_X S_X = S$. $\square$

An XP Algorithm

Theorem. EDGE DELETION TO p-PAGE PLANAR is in XP with respect to $h + p$.

An XP Algorithm

Theorem. EDGE DELETION TO p-PAGE PLANAR is in XP with respect to $h + p$.

Proof. Recall: For $q \in [p]$, $\mathcal{X}^q$ is the family of subsets of H that are bridged on page q.

An XP Algorithm

Theorem. EDGE DELETION TO p-PAGE PLANAR is in XP with respect to $h + p$.

Proof. Recall: For $q \in [p]$, $\mathcal{X}^q$ is the family of subsets of H that are bridged on page q.

Then $|\mathcal{X}^p| \leq$

An XP Algorithm

Theorem. EDGE DELETION TO p-PAGE PLANAR is in XP with respect to $h + p$.

Proof. Recall: For $q \in [p]$, $\mathcal{X}^q$ is the family of subsets of H that are bridged on page q.

Then $|\mathcal{X}^p| \leq$ × × × × × × × × × ×

An XP Algorithm

Theorem. EDGE DELETION TO p-PAGE PLANAR is in XP with respect to $h + p$.

Proof. Recall: For $q \in [p]$, $\mathcal{X}^q$ is the family of subsets of H that are bridged on page q.

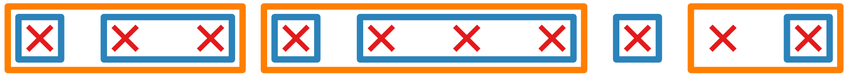
Then $|\mathcal{X}^p| \leq$

An XP Algorithm

Theorem. EDGE DELETION TO p-PAGE PLANAR is in XP with respect to $h + p$.

Proof. Recall: For $q \in [p]$, $\mathcal{X}^q$ is the family of subsets of H that are bridged on page q.

Then $|\mathcal{X}^p| \leq$

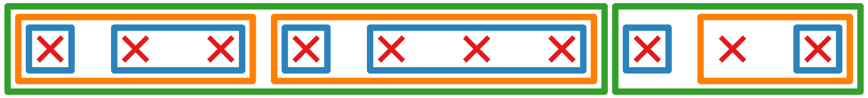


An XP Algorithm

Theorem. EDGE DELETION TO p-PAGE PLANAR is in XP with respect to $h + p$.

Proof. Recall: For $q \in [p]$, $\mathcal{X}^q$ is the family of subsets of H that are bridged on page q.

Then $|\mathcal{X}^p| \leq$

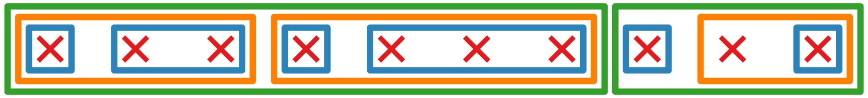


An XP Algorithm

Theorem. EDGE DELETION TO p-PAGE PLANAR is in XP with respect to $h + p$.

Proof. Recall: For $q \in [p]$, $\mathcal{X}^q$ is the family of subsets of H that are bridged on page q.

Then $|\mathcal{X}^p| \leq 2h - 1$.

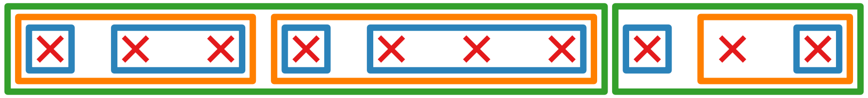


An XP Algorithm

Theorem. EDGE DELETION TO p-PAGE PLANAR is in XP with respect to $h + p$.

Proof. Recall: For $q \in [p]$, $\mathcal{X}^q$ is the family of subsets of H that are bridged on page q.

Then $|\mathcal{X}^p| \leq 2h - 1$.

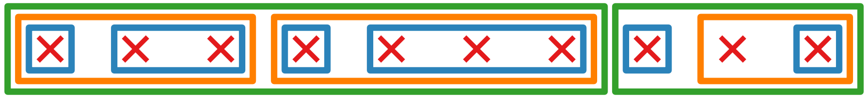


$\Rightarrow$ partial encod. $\mathcal{E}^q$ chooses $\leq 4h - 2$ edges e_X^q / f_X^q.

An XP Algorithm

Theorem. EDGE DELETION TO p-PAGE PLANAR is in XP with respect to $h + p$.

Proof. Recall: For $q \in [p]$, $\mathcal{X}^q$ is the family of subsets of H that are bridged on page q.

Then $|\mathcal{X}^p| \leq 2h - 1$. 

$\Rightarrow$ partial encod. $\mathcal{E}^q$ chooses $\leq 4h - 2$ edges e_X^q / f_X^q.

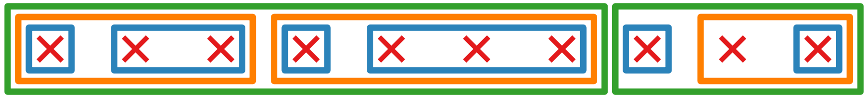
$\Rightarrow$ # encodings $\langle \mathcal{E}^1, \ldots, \mathcal{E}^p \rangle \leq$

An XP Algorithm

Theorem. EDGE DELETION TO p-PAGE PLANAR is in XP with respect to $h + p$.

Proof. Recall: For $q \in [p]$, $\mathcal{X}^q$ is the family of subsets of H that are bridged on page q.

Then $|\mathcal{X}^p| \leq 2h - 1$.



$\Rightarrow$ partial encod. $\mathcal{E}^q$ chooses $\leq 4h - 2$ edges e^q_X / f^q_X.

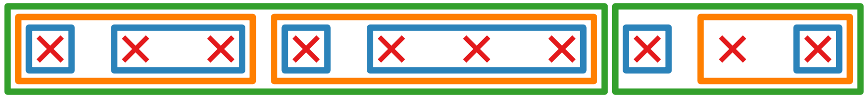
$\Rightarrow$ # encodings $\langle \mathcal{E}^1, \ldots, \mathcal{E}^p \rangle \leq m^{p \cdot (4h-2)}$

An XP Algorithm

Theorem. EDGE DELETION TO p-PAGE PLANAR is in XP with respect to $h + p$.

Proof. Recall: For $q \in [p]$, $\mathcal{X}^q$ is the family of subsets of H that are bridged on page q.

Then $|\mathcal{X}^p| \leq 2h - 1$.



$\Rightarrow$ partial encod. $\mathcal{E}^q$ chooses $\leq 4h - 2$ edges e_X^q / f_X^q.

$\Rightarrow$ # encodings $\langle \mathcal{E}^1, \ldots, \mathcal{E}^p \rangle \leq m^{p \cdot (4h-2)}$

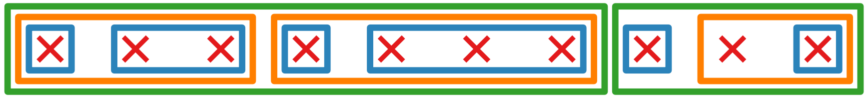
For each $X \subseteq H$, we apply Lem. 1 in $\tilde{O}\big(|E_X|^3\big)$ time.

An XP Algorithm

Theorem. EDGE DELETION TO p-PAGE PLANAR is in XP with respect to $h + p$.

Proof. Recall: For $q \in [p]$, $\mathcal{X}^q$ is the family of subsets of H that are bridged on page q.

Then $|\mathcal{X}^p| \leq 2h - 1$.



$\Rightarrow$ partial encod. $\mathcal{E}^q$ chooses $\leq 4h - 2$ edges e_X^q / f_X^q.

$\Rightarrow$ # encodings $\langle \mathcal{E}^1, \dots, \mathcal{E}^p \rangle \leq m^{p \cdot (4h-2)}$

For each $X \subseteq H$, we apply Lem. 1 in $\tilde{O}\big(|E_X|^3\big)$ time.

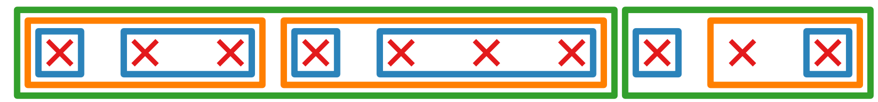
If $X \neq X'$, then $E_X \cap E_{X'} = \emptyset$.

An XP Algorithm

Theorem. EDGE DELETION TO p-PAGE PLANAR is in XP with respect to $h + p$.

Proof. Recall: For $q \in [p]$, $\mathcal{X}^q$ is the family of subsets of H that are bridged on page q.

Then $|\mathcal{X}^p| \leq 2h - 1$.



$\Rightarrow$ partial encod. $\mathcal{E}^q$ chooses $\leq 4h - 2$ edges e_X^q / f_X^q.

$\Rightarrow$ # encodings $\langle \mathcal{E}^1, \ldots, \mathcal{E}^p \rangle \leq m^{p \cdot (4h-2)}$

For each $X \subseteq H$, we apply Lem. 1 in $\tilde{O}(|E_X|^3)$ time.

If $X \neq X'$, then $E_X \cap E_{X'} = \emptyset$.

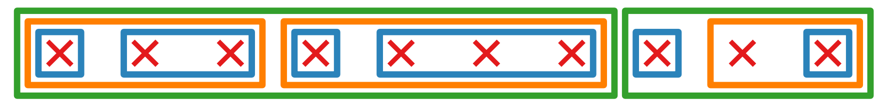
$\Rightarrow$ Per encoding, we spend $\tilde{O}(m^3)$ time (for flows).

An XP Algorithm

Theorem. EDGE DELETION TO p-PAGE PLANAR is in XP with respect to $h + p$.

Proof. Recall: For $q \in [p]$, $\mathcal{X}^q$ is the family of subsets of H that are bridged on page q.

Then $|\mathcal{X}^p| \leq 2h - 1$.



$\Rightarrow$ partial encod. $\mathcal{E}^q$ chooses $\leq 4h - 2$ edges e_X^q / f_X^q.

$\Rightarrow$ # encodings $\langle \mathcal{E}^1, \ldots, \mathcal{E}^p \rangle \leq m^{p \cdot (4h-2)}$

For each $X \subseteq H$, we apply Lem. 1 in $\tilde{O}(|E_X|^3)$ time.

If $X \neq X'$, then $E_X \cap E_{X'} = \emptyset$.

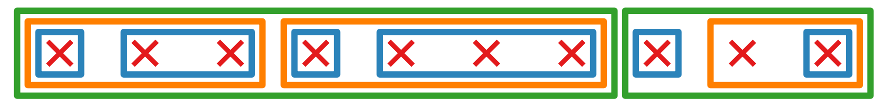
$\Rightarrow$ Per encoding, we spend $\tilde{O}(m^3)$ time (for flows).

An XP Algorithm

Theorem. EDGE DELETION TO p-PAGE PLANAR is in XP with respect to $h + p$.

Proof. Recall: For $q \in [p]$, $\mathcal{X}^q$ is the family of subsets of H that are bridged on page q.

Then $|\mathcal{X}^p| \leq 2h - 1$.



$\Rightarrow$ partial encod. $\mathcal{E}^q$ chooses $\leq 4h - 2$ edges e_X^q / f_X^q.

$\Rightarrow$ # encodings $\langle \mathcal{E}^1, \ldots, \mathcal{E}^p \rangle \leq m^{p \cdot (4h-2)}$

For each $X \subseteq H$, we apply Lem. 1 in $\tilde{O}(|E_X|^3)$ time.

If $X \neq X'$, then $E_X \cap E_{X'} = \emptyset$.

$\Rightarrow$ Per encoding, we spend $\tilde{O}(m^3)$ time (for flows).

$\Rightarrow$ Total running time is $\tilde{O}(m^{p \cdot (4h-2)+3})$. $\square$

Open Problems

- Is Edge Deletion to Page-p Planar even in FPT?

Open Problems

- Is EDGE DELETION TO PAGE-p PLANAR even in FPT?
- Is EDGE DELETION TO 1-PAGE d-PLANAR $W[1]$-hard w.r.t. the natural parameter k if d is part of the input?

Open Problems

- Is Edge Deletion to Page-p Planar even in FPT?
- Is Edge Deletion to 1-Page d-Planar $W[1]$-hard w.r.t. the natural parameter k if d is part of the input?

  Can we reduce from Independent Set?

Open Problems

- Is Edge Deletion to Page-p Planar even in FPT?

- Is Edge Deletion to 1-Page d-Planar $W[1]$-hard w.r.t. the natural parameter k if d is part of the input?

  Can we reduce from Independent Set?

  Note that Deletion to Degree-d is $W[1]$-hard with respect to treewidth [Betzler, Bredereck, Niedermeier, Uhlmann 2012] and that outer d-planar graphs have treewidth $O(d)$ [Wood & Telle, 2007]

Open Problems

- Is Edge Deletion to Page-p Planar even in FPT?
- Is Edge Deletion to 1-Page d-Planar $W[1]$-hard w.r.t. the natural parameter k if d is part of the input?

  Can we reduce from Independent Set?

  Note that Deletion to Degree-d is $W[1]$-hard with respect to treewidth [Betzler, Bredereck, Niedermeier, Uhlmann 2012] and that outer d-planar graphs have treewidth $O(d)$ [Wood & Telle, 2007]
- Can the fixed-order crossing number be computed in $2^{n} n^{O(1)}$ instead of $2^{m} n^{O(1)}$ time?

Open Problems

- Is Edge Deletion to Page-p Planar even in FPT?
- Is Edge Deletion to 1-Page d-Planar $W[1]$-hard w.r.t. the natural parameter k if d is part of the input?

  Can we reduce from Independent Set?

  Note that Deletion to Degree-d is $W[1]$-hard with respect to treewidth [Betzler, Bredereck, Niedermeier, Uhlmann 2012] and that outer d-planar graphs have treewidth $O(d)$ [Wood & Telle, 2007]
- Can the fixed-order crossing number be computed in $2^{n} n^{O(1)}$ instead of $2^{m} n^{O(1)}$ time?
- What is the parameterized complexity of Edge Deletion to Outer d-Planarity (that is, for unordered graphs)?