Fully Scalable MPC Algorithms for Embedded Planar Graphs

Graph drawing for planar graph algorithms

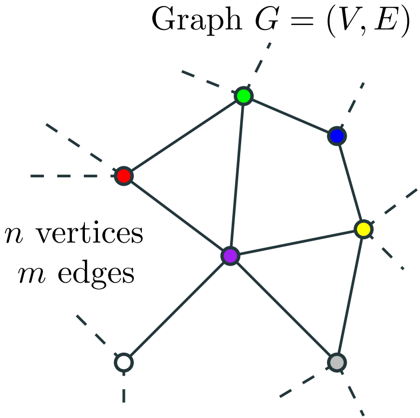
Yi-Jun Chang (NUS) **Da Wei Zheng (UIUC)**

Jan 29, 2024

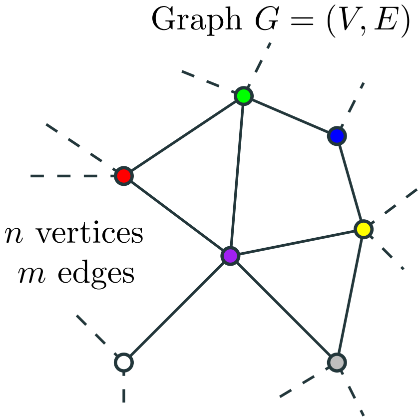
UIUC Theory Seminar

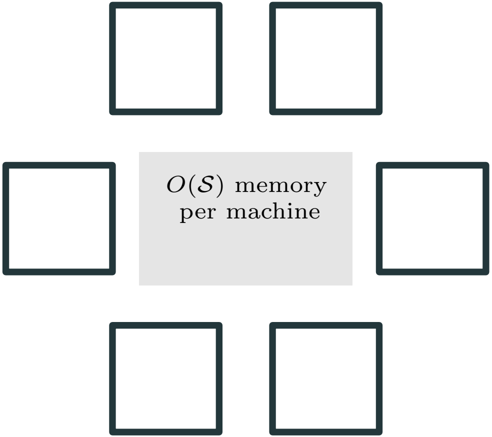
Presented at SODA 2024

Massively parallel computing with graphs

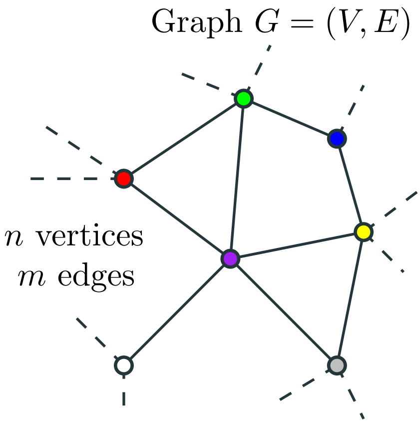


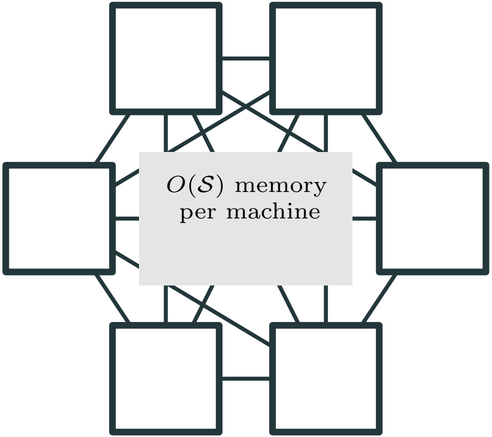
Massively parallel computing with graphs



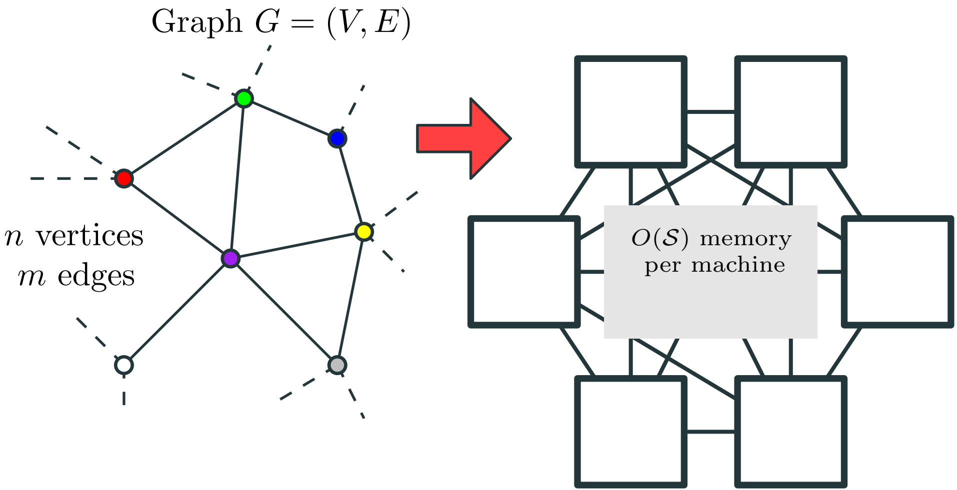


Massively parallel computing with graphs

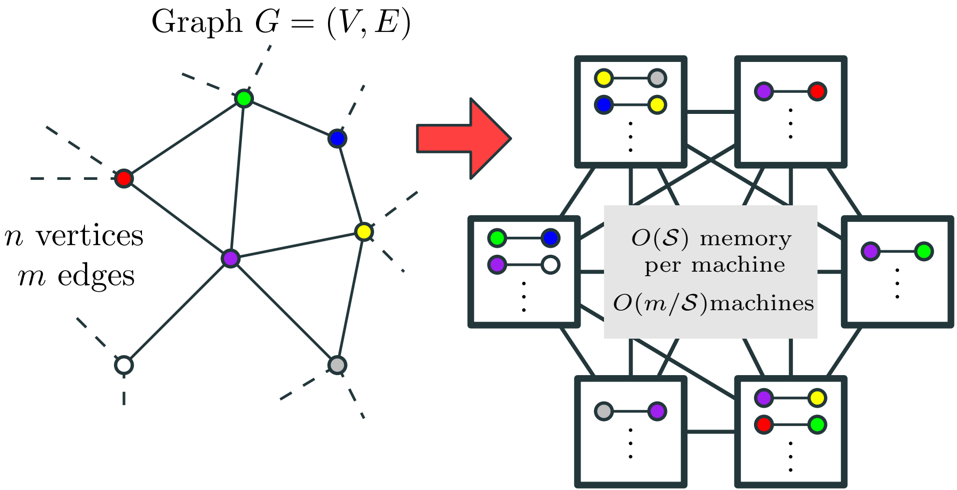




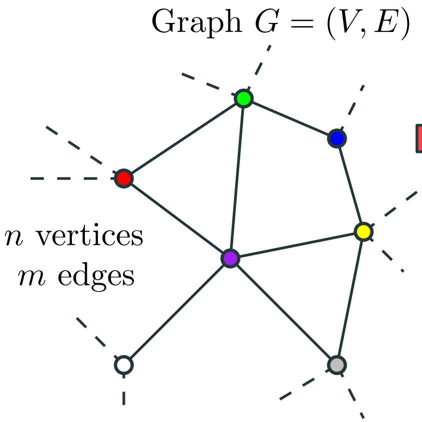
Massively parallel computing with graphs

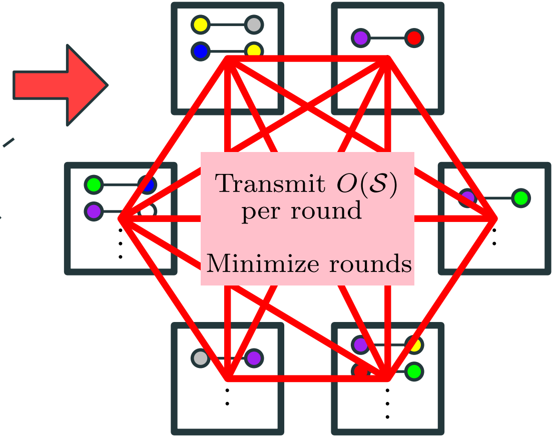


Massively parallel computing with graphs

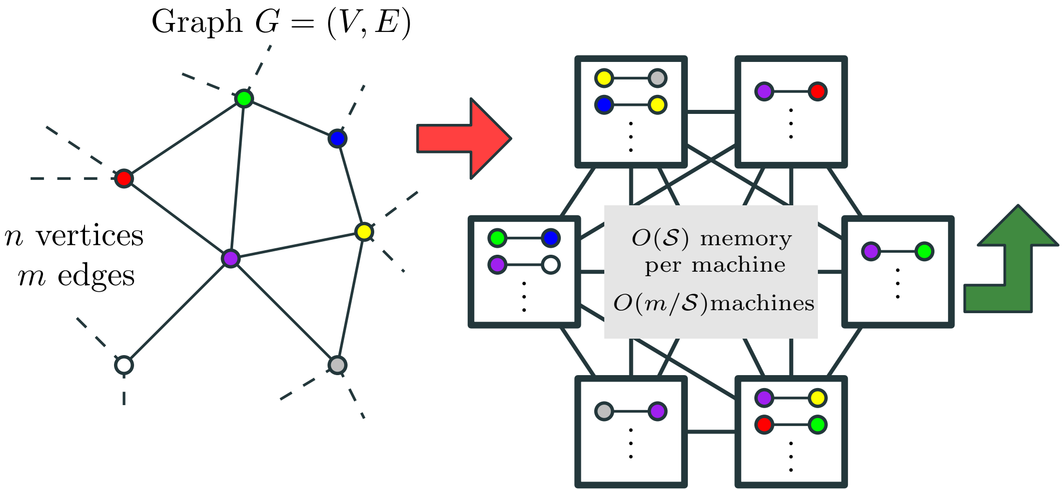


Massively parallel computing with graphs

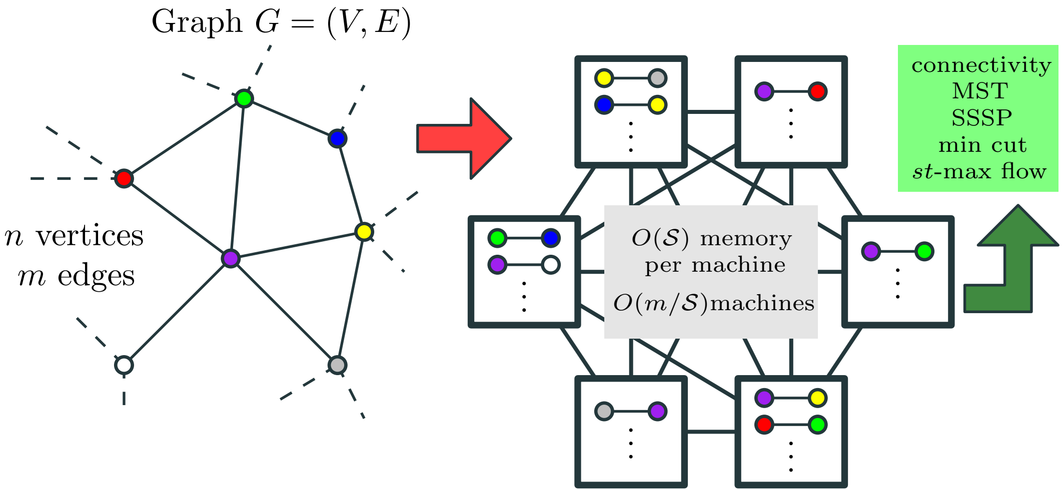




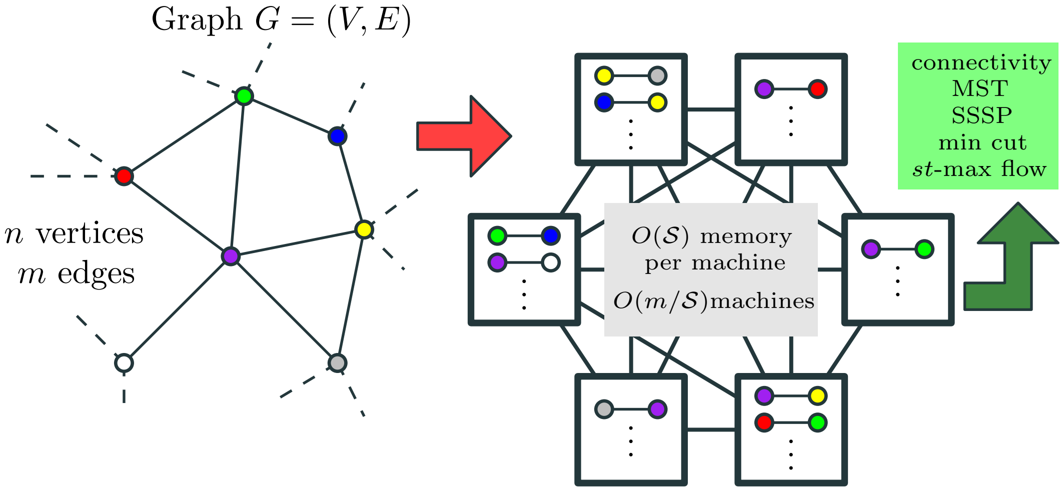
Massively parallel computing with graphs



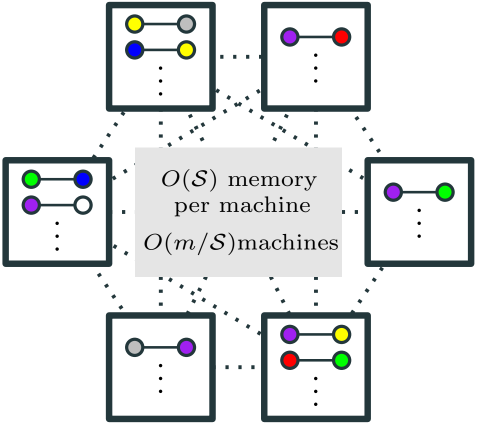
Massively parallel computing with graphs

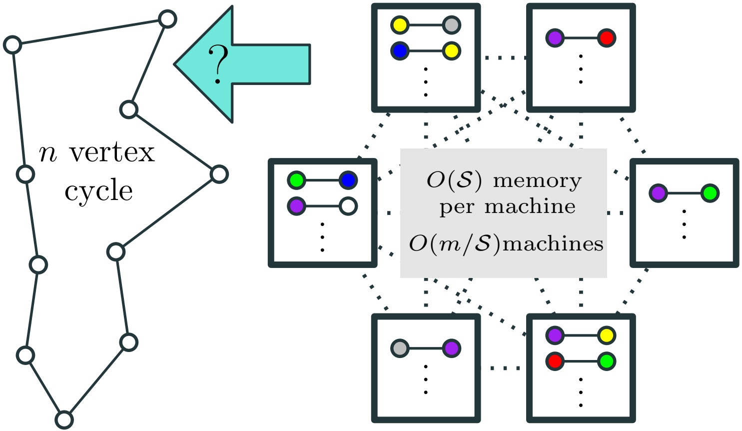


Definition
We say an algorithm is **fully scalable** if it works with $\mathcal{S} = n^{\delta}$ for any $\delta > 0$.

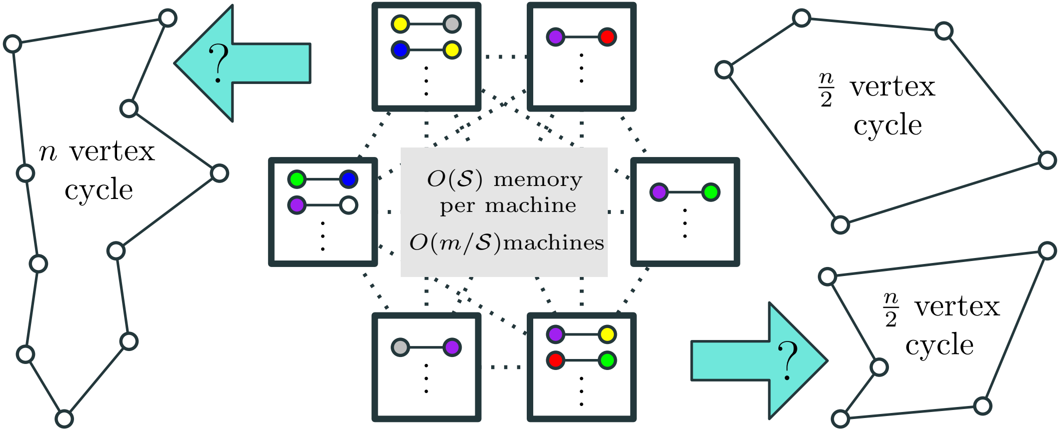


A bottleneck: The 1-vs-2 cycle conjecture



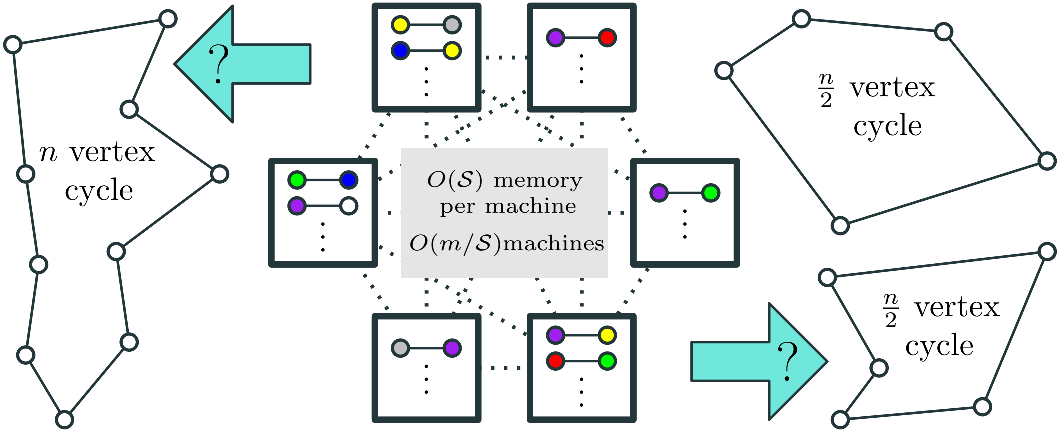

n vertex
cycle
?
$O(\mathcal{S})$ memory
per machine
$O(m/\mathcal{S})$machines

A bottleneck: The 1-vs-2 cycle conjecture



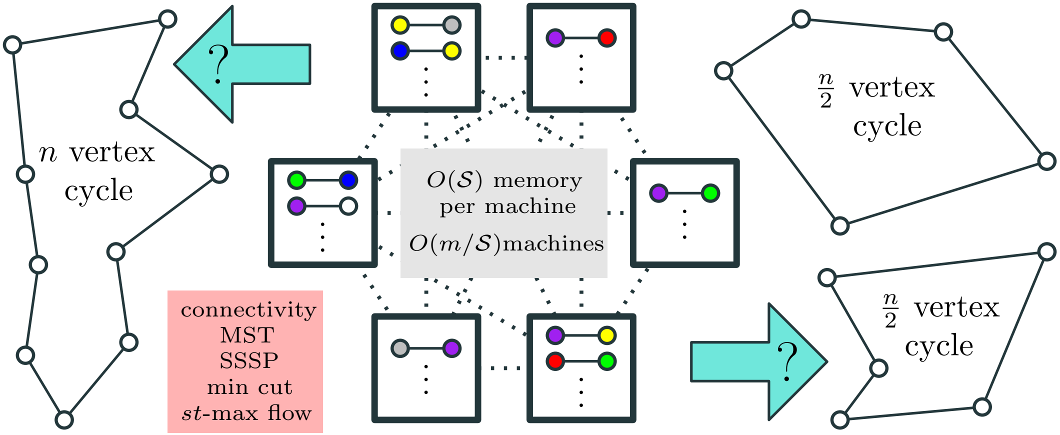
A bottleneck: The 1-vs-2 cycle conjecture

If $\mathcal{S} = O(n^{1-\varepsilon})$, then need $\Omega(\log n)$ rounds.



A bottleneck: The 1-vs-2 cycle conjecture

If $\mathcal{S} = O(n^{1-\varepsilon})$, then need $\Omega(\log n)$ rounds.



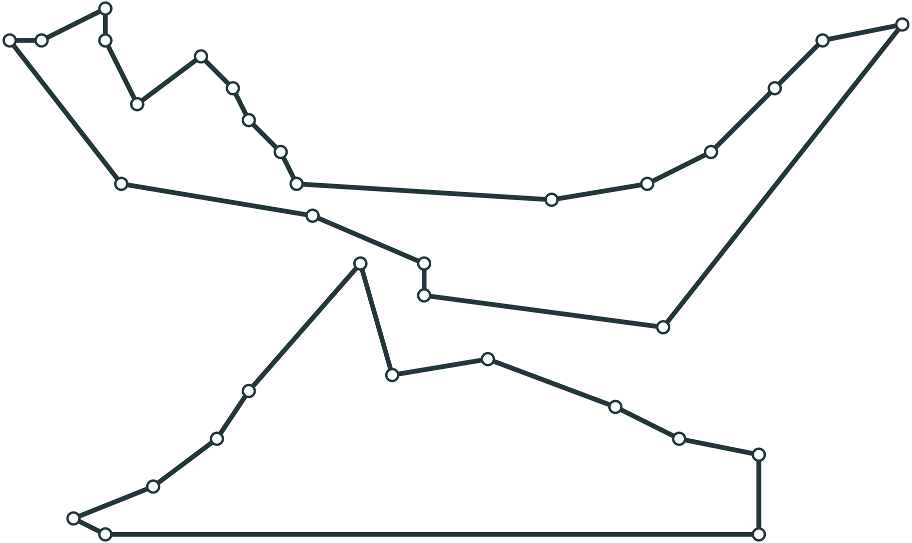
This paper:

$O(1)$-**round**

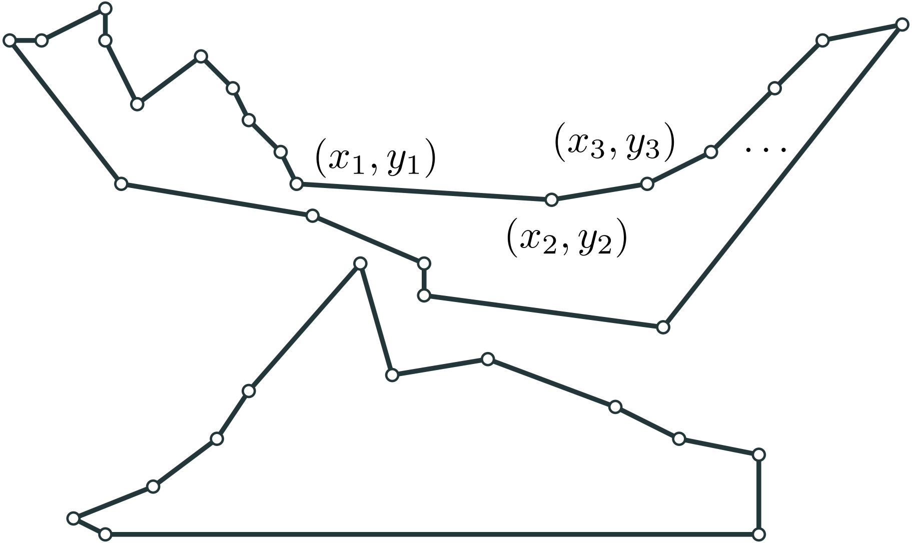
fully scalable

algorithms!

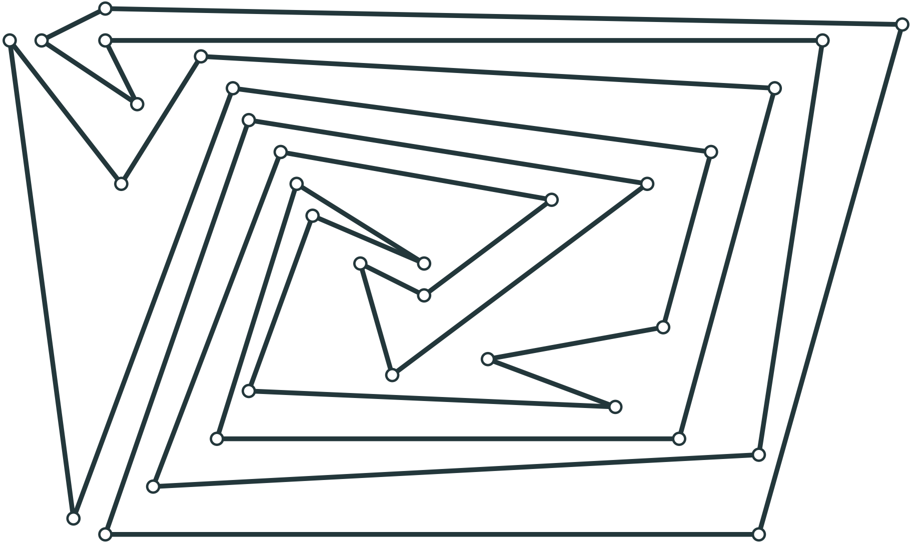
Embedded planar graphs



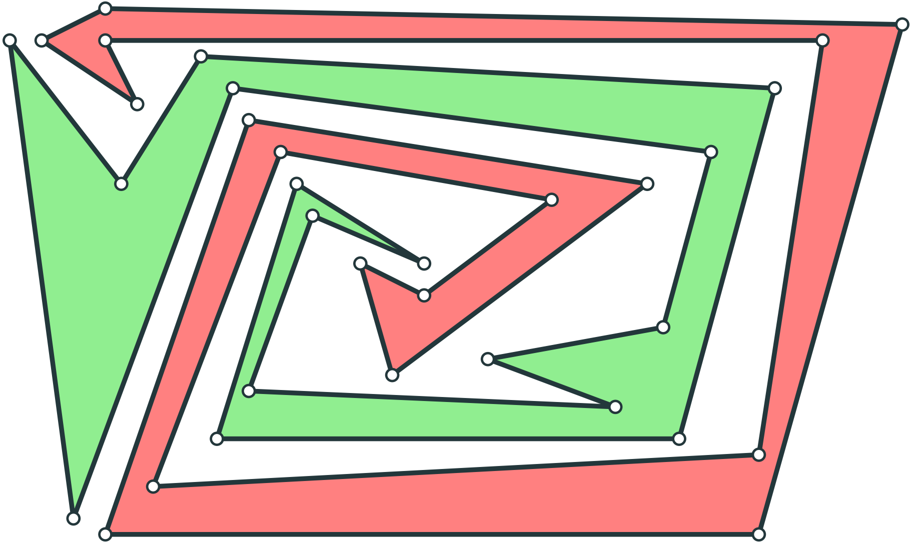
Embedded planar graphs



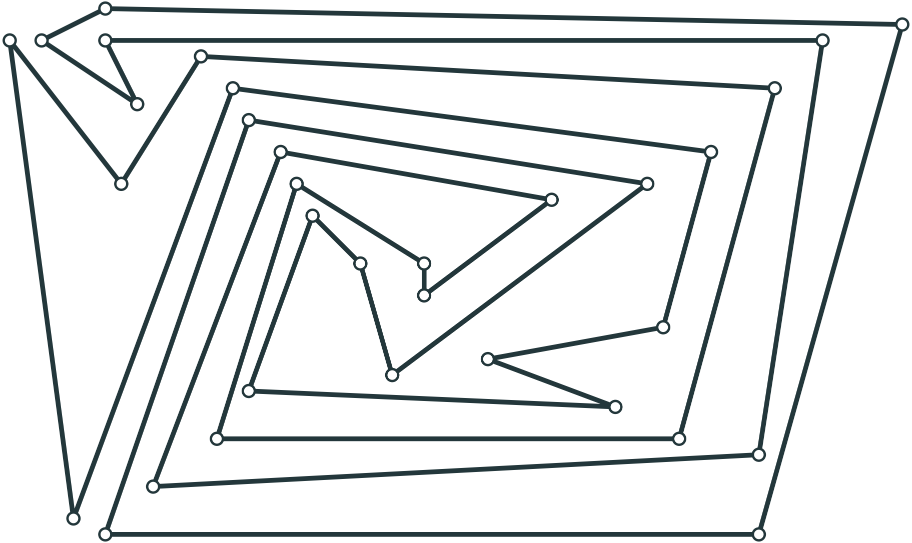
Embedded planar graphs



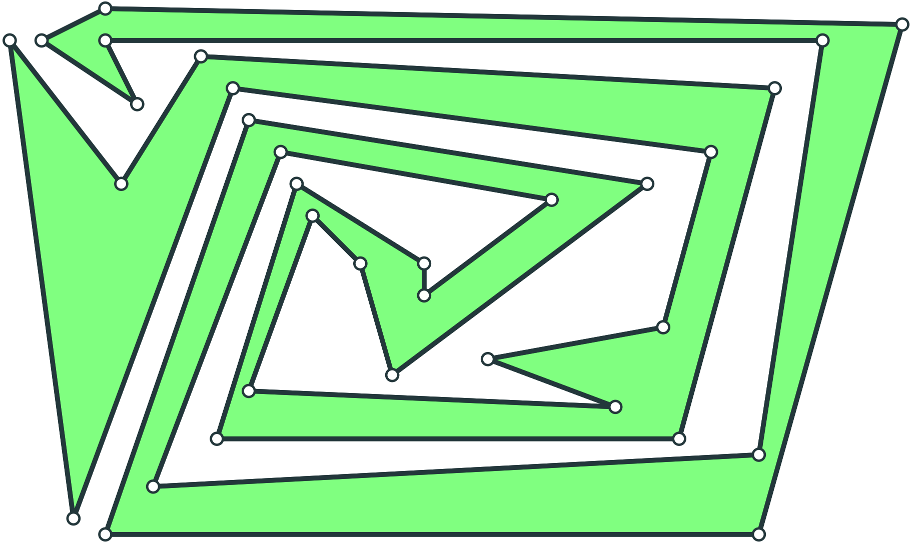
Embedded planar graphs



Embedded planar graphs



Embedded planar graphs



Overcoming the 1-vs-2 cycle conjecture

	Problem	Total space	Memory per machine	Source
Embedded planar graphs	Connected Components	$O(n)$	$n^{2/3+\Omega(1)}$	[HT23]
	Minimum Spanning Tree	$O(n)$	$n^{2/3+\Omega(1)}$	[HT23]
	$O(1)$-approx. SSSP	$O(n)$	$n^{2/3+\Omega(1)}$	[HT23]

Overcoming the 1-vs-2 cycle conjecture

	Problem	Total space	Memory per machine	Source
Embedded planar graphs	Connected Components	$O(n)$	$n^{2/3+\Omega(1)}$	[HT23]
		$O(n)$	n^{δ}	New!
	Minimum Spanning Tree	$O(n)$	$n^{2/3+\Omega(1)}$	[HT23]
		$O(n)$	n^{δ}	New!
	$O(1)$-approx. SSSP	$O(n)$	$n^{2/3+\Omega(1)}$	[HT23]
	$(1+\varepsilon)$-approx. SSSP	$O(n)$	n^{δ}	New!

Overcoming the 1-vs-2 cycle conjecture

	Problem	Total space	Memory per machine	Source
Embedded planar graphs	Connected Components	$O(n)$	$n^{2/3+\Omega(1)}$	[HT23]
		$O(n)$	n^{δ}	New!
	Minimum Spanning Tree	$O(n)$	$n^{2/3+\Omega(1)}$	[HT23]
		$O(n)$	n^{δ}	New!
	$O(1)$-approx. SSSP	$O(n)$	$n^{2/3+\Omega(1)}$	[HT23]
	$(1+\varepsilon)$-approx. SSSP	$O(n)$	n^{δ}	New!
	$(1+\varepsilon)$-approx. APSP	$O(n^2)$	n^{δ}	New!
	$(1+\varepsilon)$-approx. global min cut	$O(n)$	n^{δ}	New!
	$(1+\varepsilon)$-approx. *st*-max flow	$O(n)$	n^{δ}	New!

Beyond planar graphs

	Problem	Total space	Memory per machine	Source
2D Euclidean MST	$(1+\varepsilon)$-approx.	$O(n)$	n^{δ}	[ANOY14]
	Exact	$O(n)$	$n^{2/3+\Omega(1)}$	[HT23]

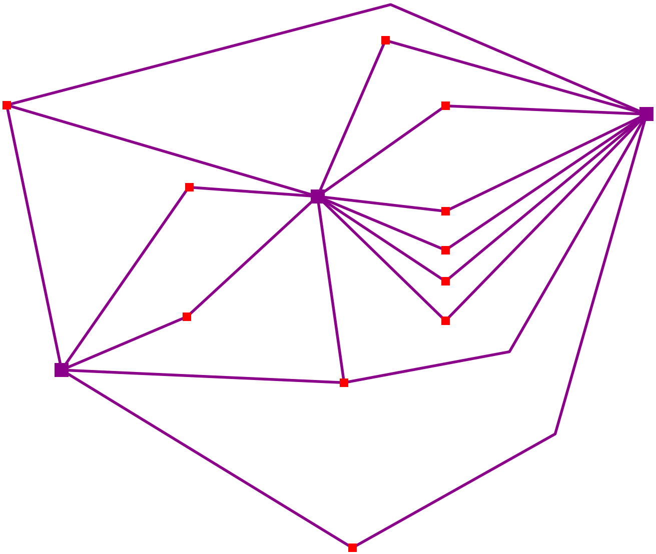
Beyond planar graphs

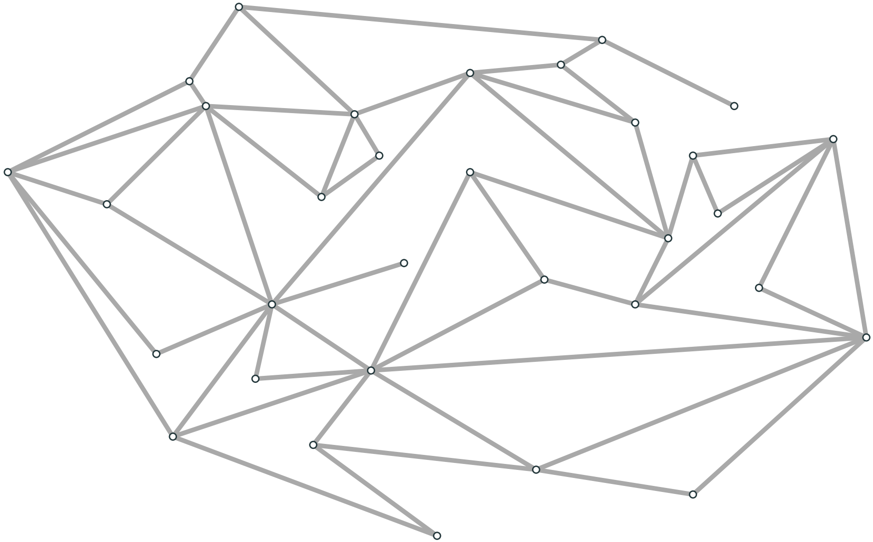
	Problem	Total space	Memory per machine	Source
2D Euclidean MST	$(1+\varepsilon)$-approx.	$O(n)$	n^{δ}	[ANOY14]
	Exact	$O(n)$	$n^{2/3+\Omega(1)}$	[HT23]
	Exact	$O(n)$	n^{δ}	New!

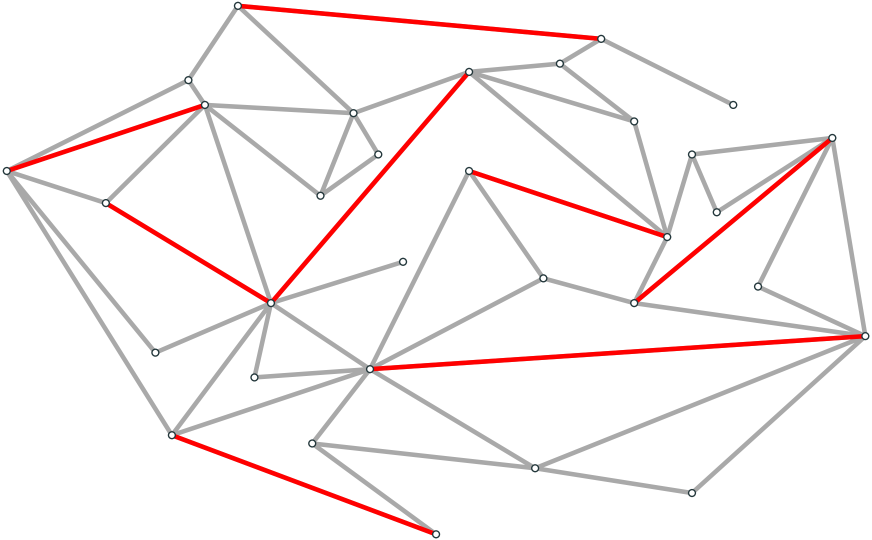
Beyond planar graphs

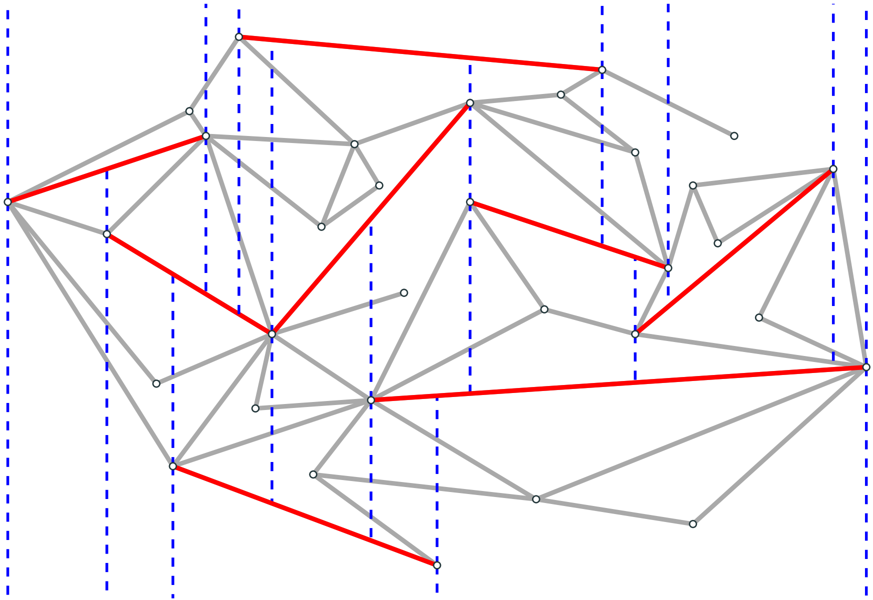
	Problem	Total space	Memory per machine	Source
2D Euclidean MST	$(1+\varepsilon)$-approx.	$O(n)$	n^{δ}	[ANOY14]
	Exact	$O(n)$	$n^{2/3+\Omega(1)}$	[HT23]
	Exact	$O(n)$	n^{δ}	New!
Edit Distance	$(3+\varepsilon)$-approx.	$\widetilde{O}(n^{(9-4\delta)/5})$	n^{δ}	[BGS21]
	$(1+\varepsilon)$-approx.	$\widetilde{O}(n^{2-\delta})$	n^{δ}	[HSS19]
	$(1+\varepsilon)$-approx. weighted	$\widetilde{O}(n^{2-\delta})$	n^{δ}	New!

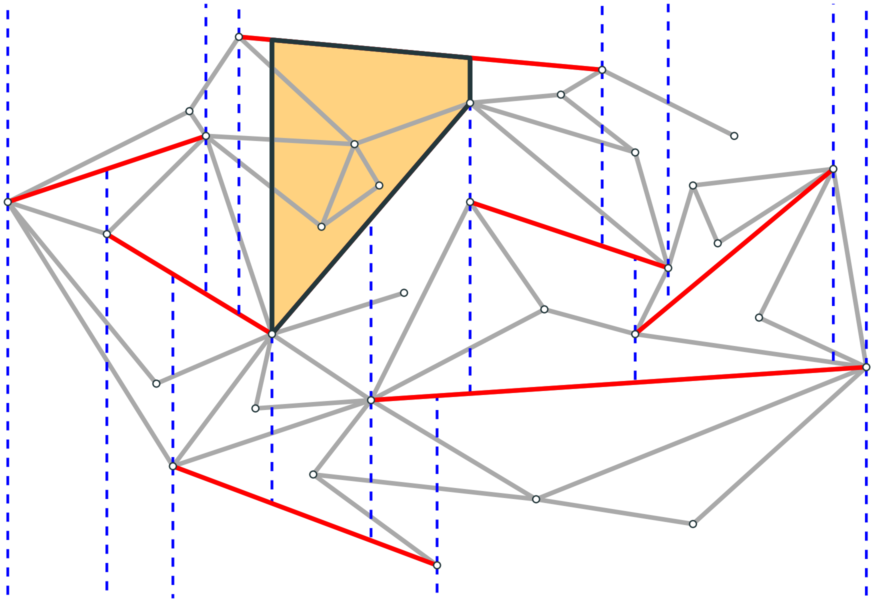
The framework of Holm and Tětek







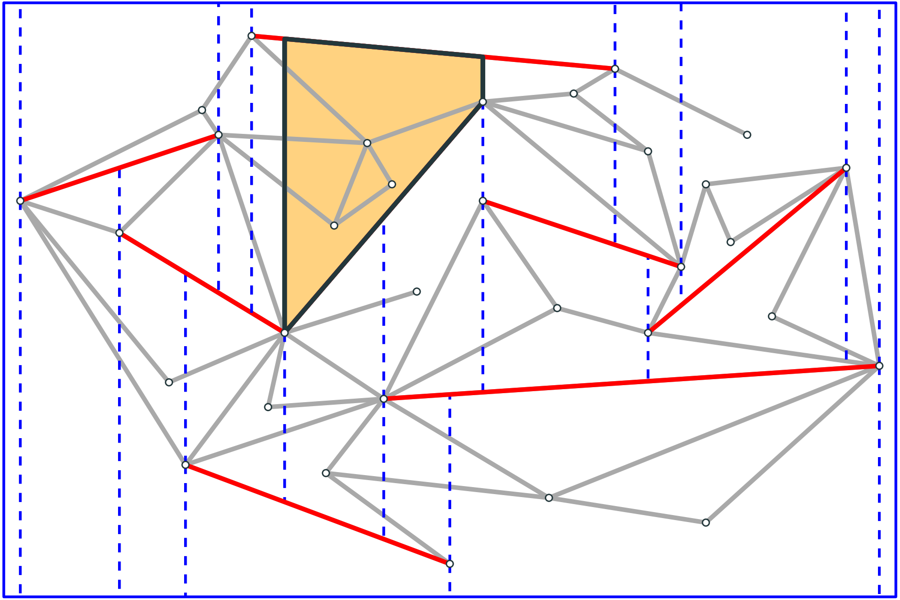




$(1/r)$**-cutting**

$O(r)$ trapezoids

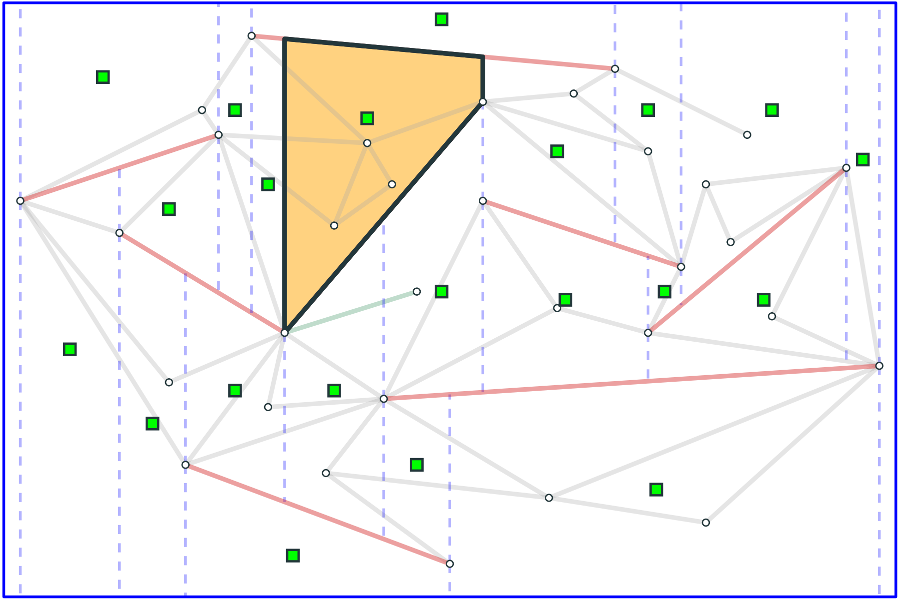
n/r edges / trap.



$(1/r)$-cutting

$O(r)$ trapezoids

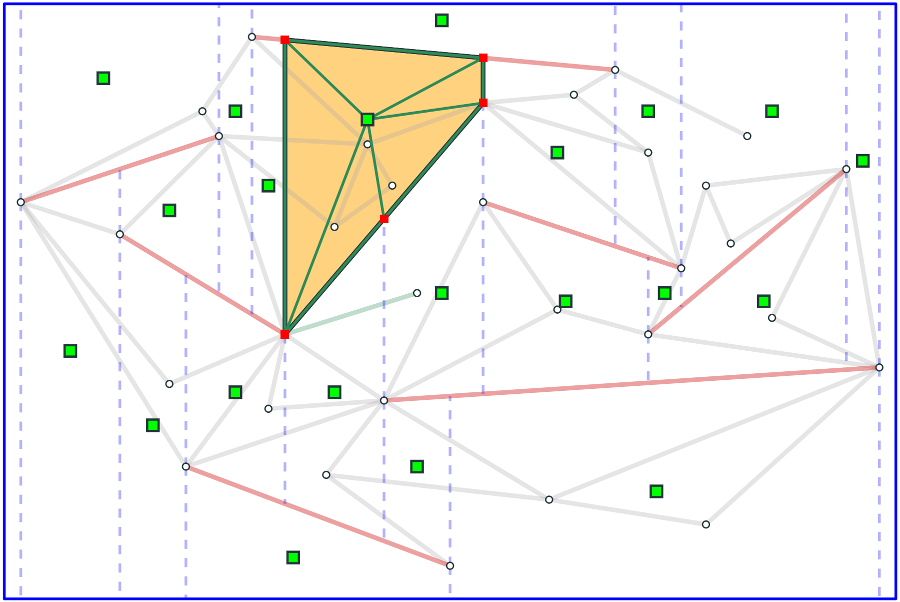
n/r edges / trap.



$(1/r)$-cutting

$O(r)$ trapezoids

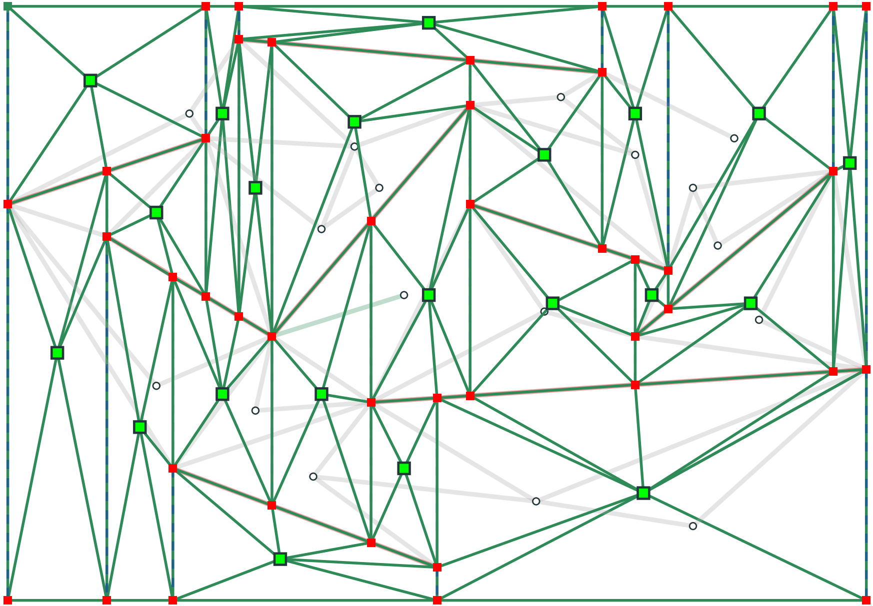
n/r edges / trap.



$(1/r)$-cutting

$O(r)$ trapezoids

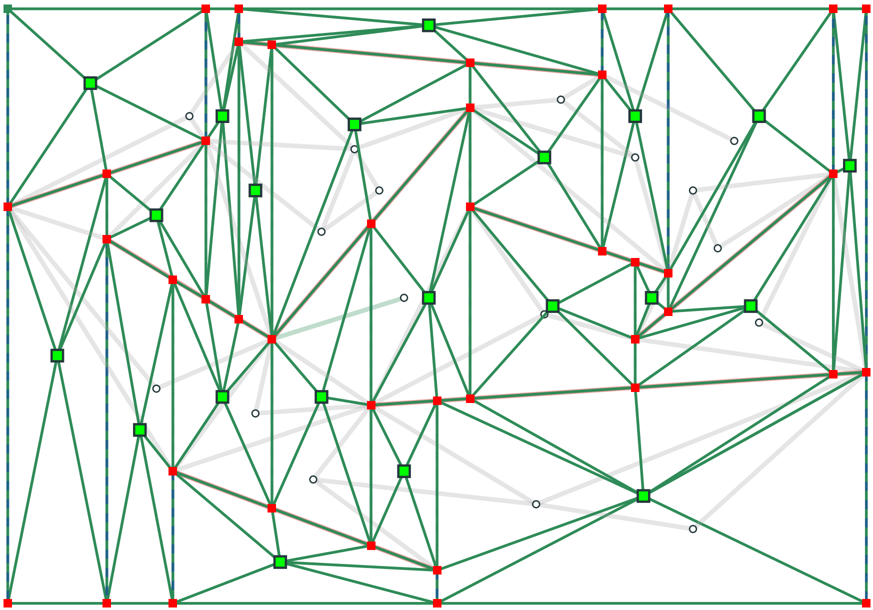
n/r edges / trap.



$(1/r)$-cutting

$O(r)$ trapezoids

n/r edges / trap.



(1/r)-cutting

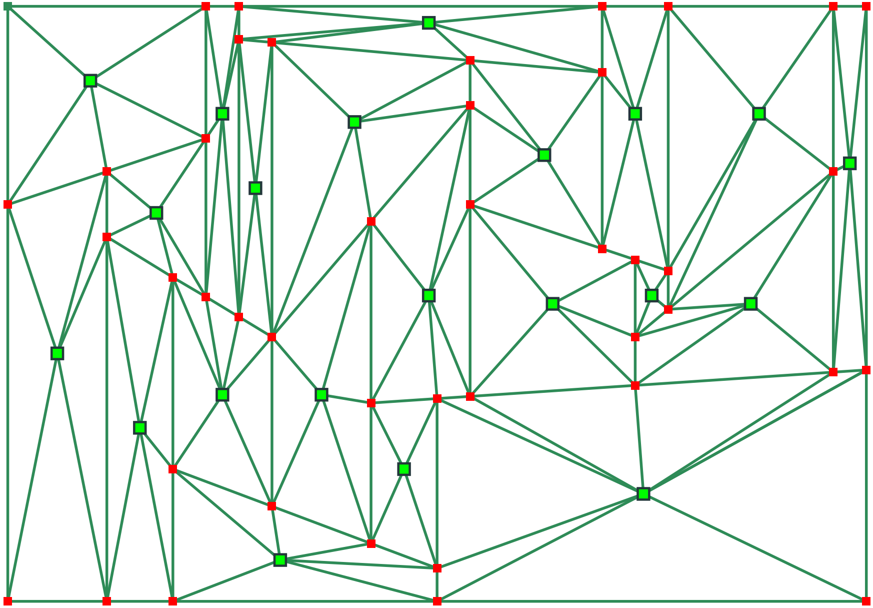
$O(r)$ trapezoids

n/r edges / trap.

Cutting graph

Planar

$O(r)$ vert. and edges



$(1/r)$-cutting

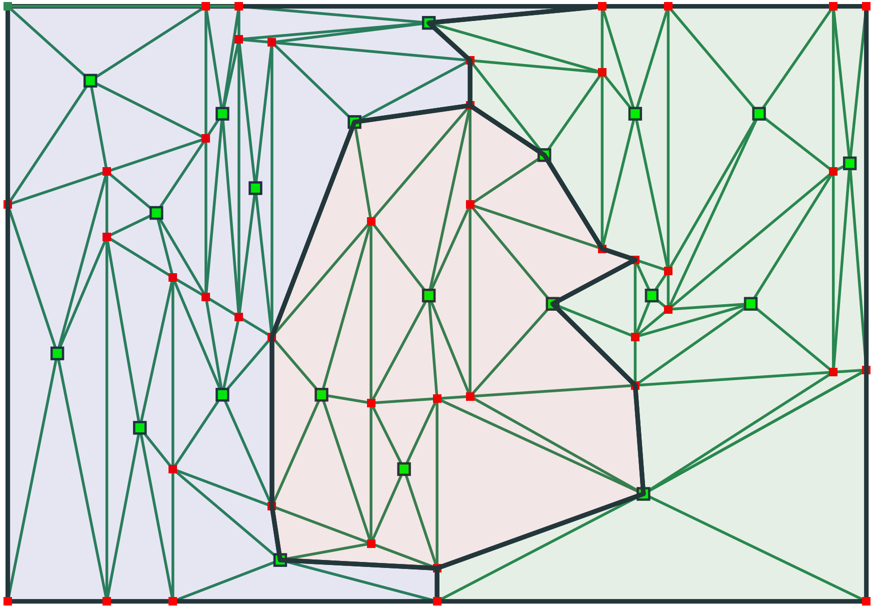
$O(r)$ trapezoids

n/r edges / trap.

Cutting graph

Planar

$O(r)$ vert. and edges



$(1/r)$-cutting

$O(r)$ trapezoids

n/r edges / trap.

Cutting graph

Planar

$O(r)$ vert. and edges

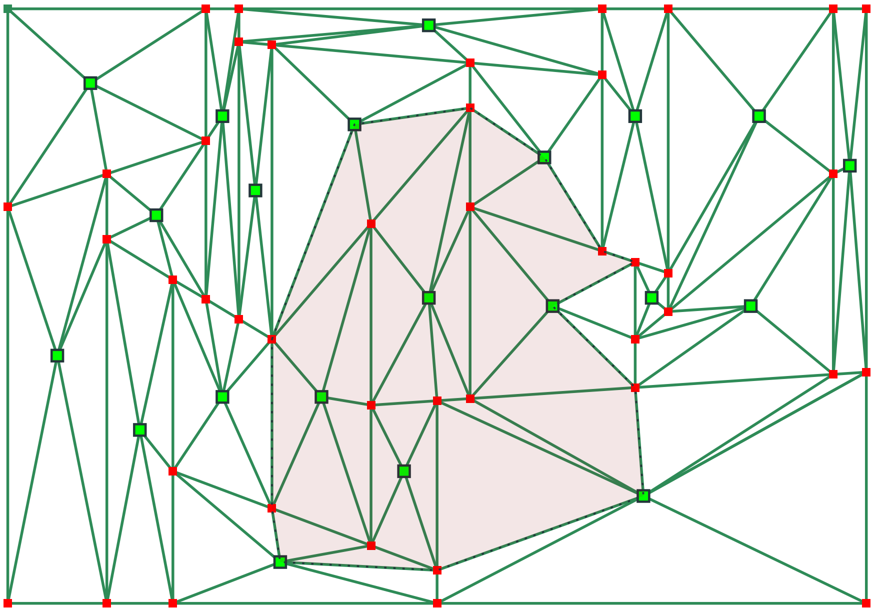
Planar s-divisions

$O(r/s)$ regions

$O(s)$ vert. / region

$O(\sqrt{s})$ vert. / bound.

$O(1)$ holes



$(1/r)$-cutting

$O(r)$ trapezoids

n/r edges / trap.

Cutting graph

Planar

$O(r)$ vert. and edges

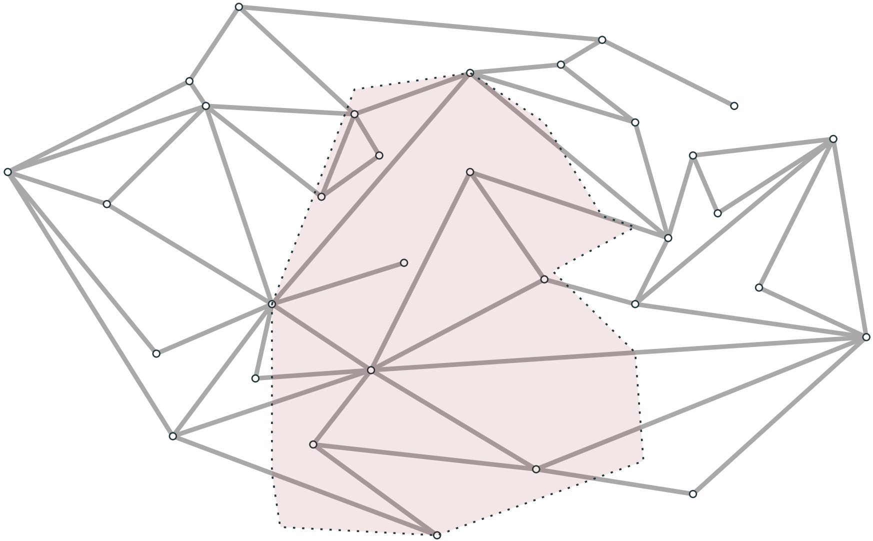
Planar s-divisions

$O(r/s)$ regions

$O(s)$ vert. / region

$O(\sqrt{s})$ vert. / bound.

$O(1)$ holes



$(1/r)$-cutting

$O(r)$ trapezoids

n/r edges / trap.

Cutting graph

Planar

$O(r)$ vert. and edges

Planar s-divisions

$O(r/s)$ regions

$O(s)$ vert. / region

$O(\sqrt{s})$ vert. / bound.

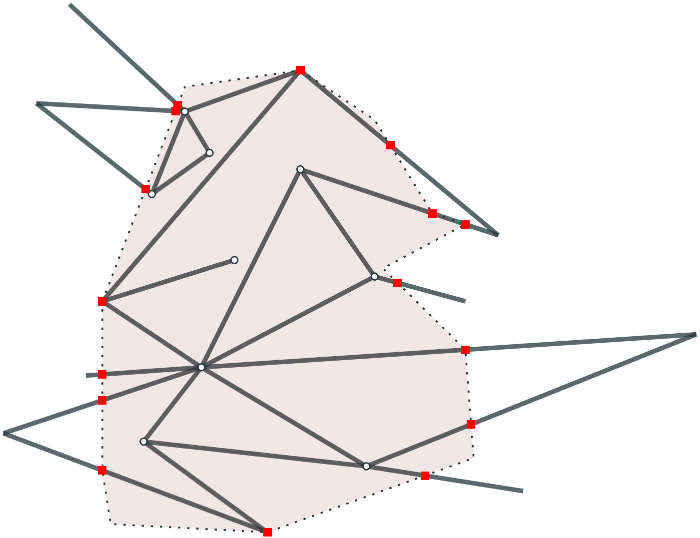
$O(1)$ holes

Cutting-division

$O(r/s)$ regions

$O\left(\frac{ns}{r}\right)$ edges / region

$O\left(\frac{n\sqrt{s}}{r}\right)$ edges / bound.



$(1/r)$-cutting

$O(r)$ trapezoids

n/r edges / trap.

Cutting graph

Planar

$O(r)$ vert. and edges

Planar s-divisions

$O(r/s)$ regions

$O(s)$ vert. / region

$O(\sqrt{s})$ vert. / bound.

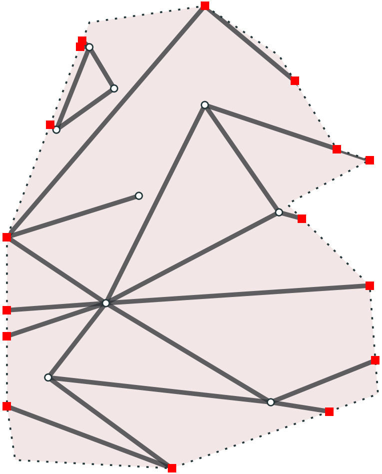
$O(1)$ holes

Cutting-division

$O(r/s)$ regions

$O\left(\frac{ns}{r}\right)$ edges / region

$O\left(\frac{n\sqrt{s}}{r}\right)$ edges / bound.



$(1/r)$-cutting

$O(r)$ trapezoids

n/r edges / trap.

Cutting graph

Planar

$O(r)$ vert. and edges

Planar s-divisions

$O(r/s)$ regions

$O(s)$ vert. / region

$O(\sqrt{s})$ vert. / bound.

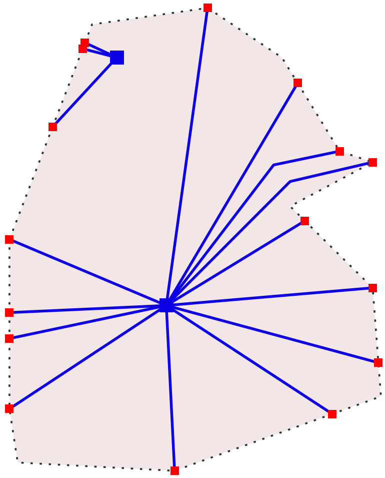
$O(1)$ holes

Cutting-division

$O(r/s)$ regions

$O\left(\frac{ns}{r}\right)$ edges / region

$O\left(\frac{n\sqrt{s}}{r}\right)$ edges / bound.



$(1/r)$-cutting

$O(r)$ trapezoids

n/r edges / trap.

Cutting graph

Planar

$O(r)$ vert. and edges

Planar s-divisions

$O(r/s)$ regions

$O(s)$ vert. / region

$O(\sqrt{s})$ vert. / bound.

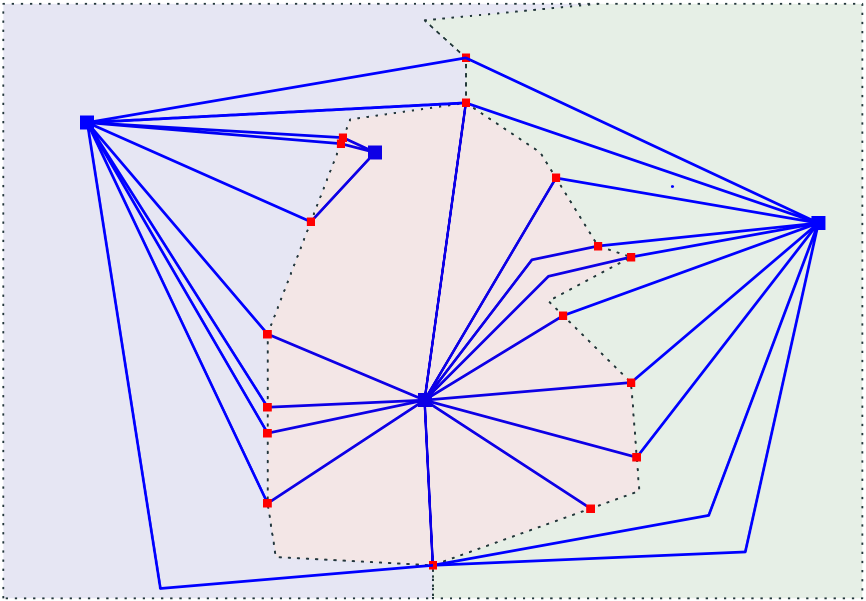
$O(1)$ holes

Cutting-division

$O(r/s)$ regions

$O\left(\frac{ns}{r}\right)$ edges / region

$O\left(\frac{n\sqrt{s}}{r}\right)$ edges / bound.



$(1/r)$-cutting

$O(r)$ trapezoids

n/r edges / trap.

Cutting graph

Planar

$O(r)$ vert. and edges

Planar s-divisions

$O(r/s)$ regions

$O(s)$ vert. / region

$O(\sqrt{s})$ vert. / bound.

$O(1)$ holes

Cutting-division

$O(r/s)$ regions

$O\left(\frac{ns}{r}\right)$ edges / region

$O\left(\frac{n\sqrt{s}}{r}\right)$ edges / bound.

Picking the parameters - Take 1 [Holm Tětek '23]

$O(r/s)$ regions $O\left(\frac{ns}{r}\right)$ edges / region $O\left(\frac{n\sqrt{s}}{r}\right)$ edges / boundary

Picking the parameters - Take 1 [Holm Tětek '23]

$O(r/s)$ regions $O\left(\frac{ns}{r}\right)$ edges / region $O\left(\frac{n\sqrt{s}}{r}\right)$ edges / boundary

Picking the parameters - Take 1 [Holm Tětek '23]

$O(r/s)$ regions $\quad$ $O\left(\frac{ns}{r}\right)$ edges / region $\quad$ $O\left(\frac{n\sqrt{s}}{r}\right)$ edges / boundary

- Local space usage for $(1/r)$-cutting: $O(r)$.

Picking the parameters - Take 1 [Holm Tětek '23]

$O(r/s)$ regions $\quad O\left(\frac{ns}{r}\right)$ edges / region $\quad O\left(\frac{n\sqrt{s}}{r}\right)$ edges / boundary

- Local space usage for $(1/r)$-cutting: $O(r)$.
- Local space usage for region after s-division: $O(ns/r)$.

Picking the parameters - Take 1 [Holm Tětek '23]

$O(r/s)$ regions $\quad$ $O\left(\frac{ns}{r}\right)$ edges / region $\quad$ $O\left(\frac{n\sqrt{s}}{r}\right)$ edges / boundary

- Local space usage for $(1/r)$-cutting: $O(r)$.
- Local space usage for region after s-division: $O(ns/r)$.
- Local space usage for after merging boundary graphs: $O(n/\sqrt{s})$.

Picking the parameters - Take 1 [Holm Tětek '23]

$O(r/s)$ regions $O\left(\frac{ns}{r}\right)$ edges / region $O\left(\frac{n\sqrt{s}}{r}\right)$ edges / boundary

- Local space usage for $(1/r)$-cutting: $O(r)$.
- Local space usage for region after s-division: $O(ns/r)$.
- Local space usage for after merging boundary graphs: $O(n/\sqrt{s})$.

Space usage: $O(ns/r + n/\sqrt{s} + r)$. Choose $s = r^{2/3}$ and $r = n^{1/2}$ for $\mathcal{S} = O(n^{3/4})$.

Picking the parameters - Take 1 [Holm Tětek '23]

$O(r/s)$ regions $O\left(\frac{ns}{r}\right)$ edges / region $O\left(\frac{n\sqrt{s}}{r}\right)$ edges / boundary

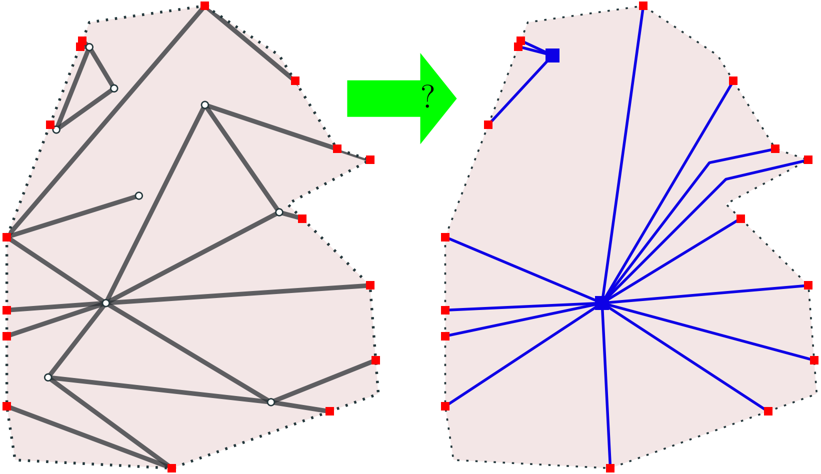
- Local space usage for $(1/r)$-cutting: $O(r)$.
- Local space usage for region after s-division: $O(ns/r)$.
- Local space usage for after merging boundary graphs: $O(n/\sqrt{s})$.

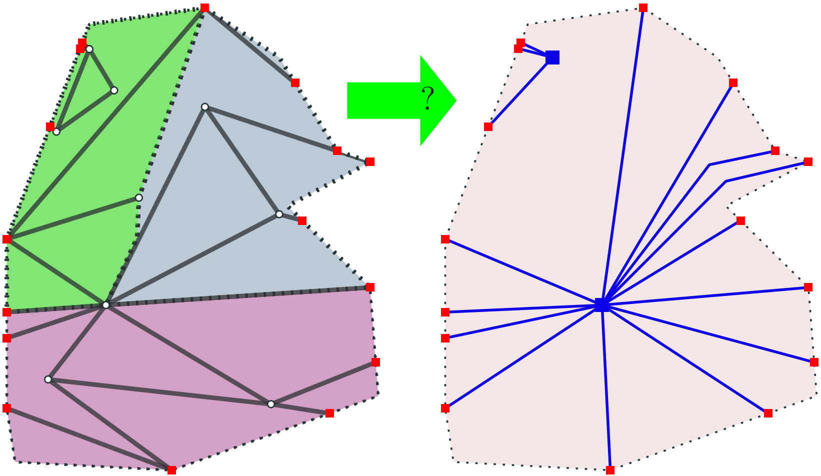
Space usage: $O(ns/r + n/\sqrt{s} + r)$. Choose $s = r^{2/3}$ and $r = n^{1/2}$ for $\mathcal{S} = O(n^{3/4})$.

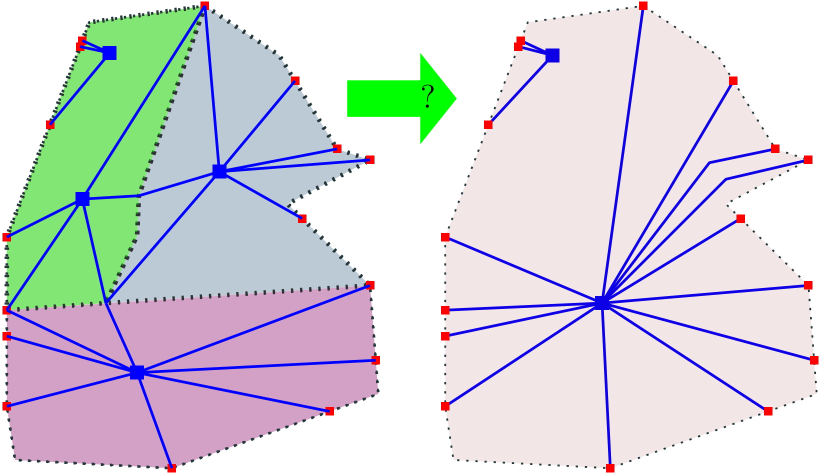
Can improve to $S = n^{2/3+\Omega(1)}$ by using some recursion.

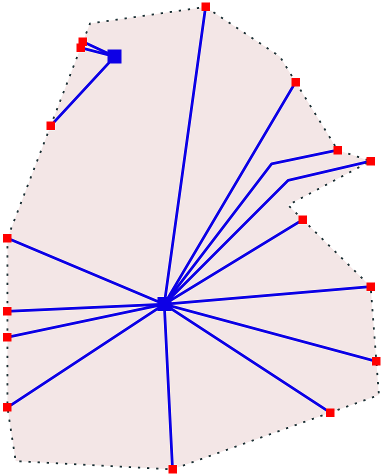
New framework:

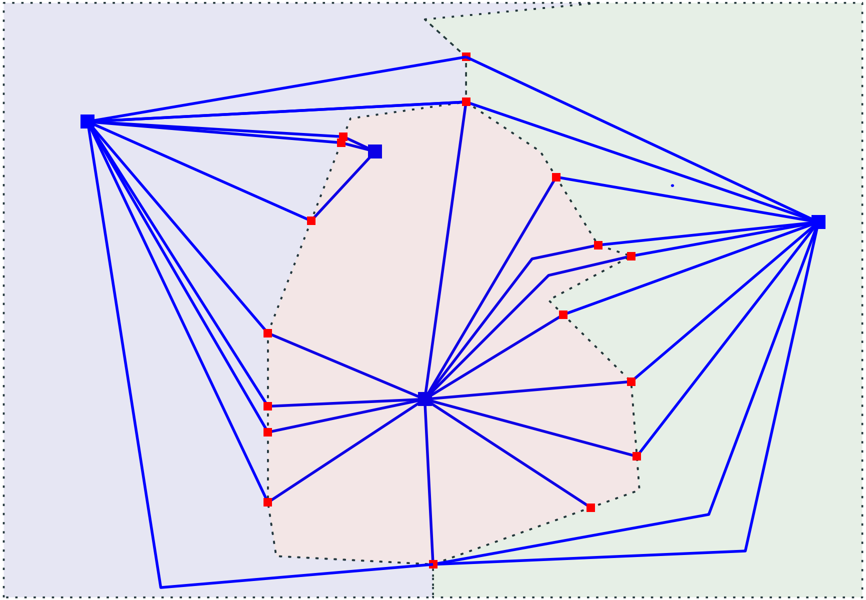
New framework:
More recursion + graph drawing!

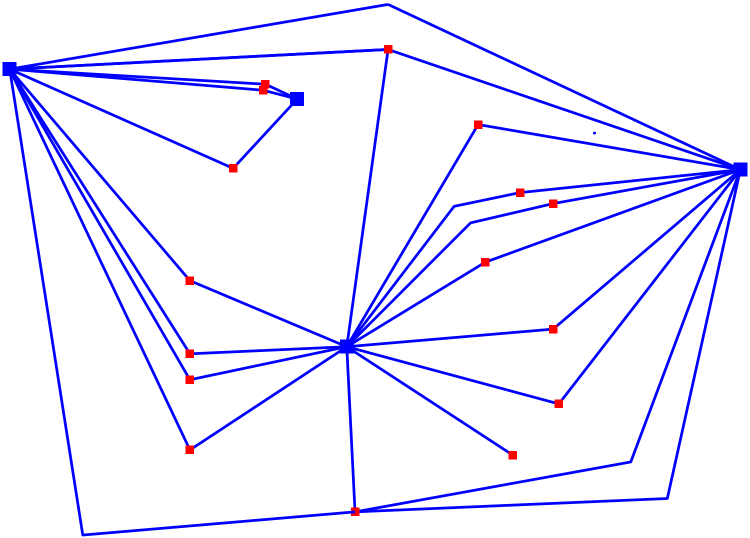


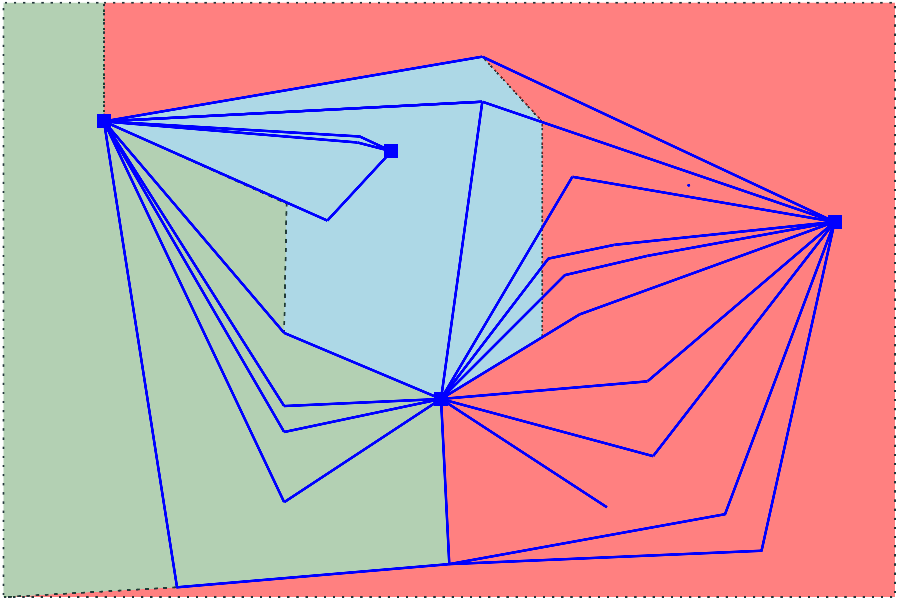


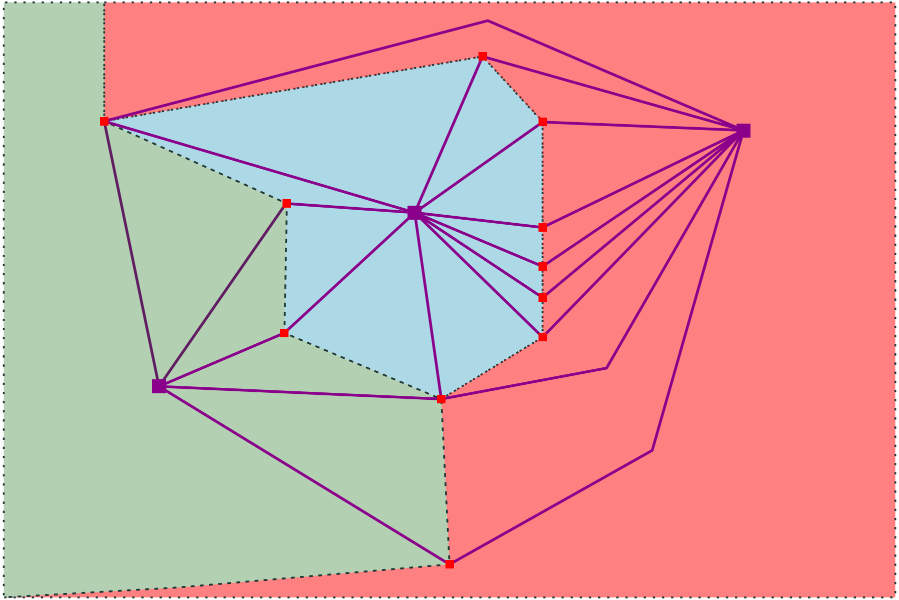


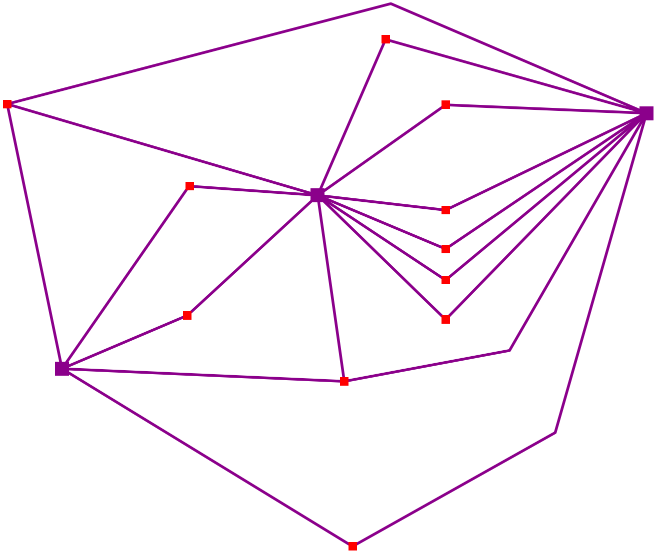












Picking the parameters - Take 2 ($\mathcal{S} = n^\delta$)

$O(r/s)$ regions $O\left(\frac{ns}{r}\right)$ edges / region $O\left(\frac{n\sqrt{s}}{r}\right)$ edges / boundary

Picking the parameters - Take 2 ($\mathcal{S} = n^\delta$)

$O(r/s)$ regions $\quad$ $O\left(\frac{ns}{r}\right)$ edges / region $\quad$ $O\left(\frac{n\sqrt{s}}{r}\right)$ edges / boundary

- Need to set $r \leq \mathcal{S} = n^\delta$ in order to compute planar s-division (on one machine)

Picking the parameters - Take 2 ($\mathcal{S} = n^\delta$)

$O(r/s)$ regions $\quad$ $O\left(\frac{ns}{r}\right)$ edges / region $\quad$ $O\left(\frac{n\sqrt{s}}{r}\right)$ edges / boundary

- Need to set $r \leq \mathcal{S} = n^\delta$ in order to compute planar s-division (on one machine)
- Need to set $s \leq r$, choose $s = r^{2/3} = n^{2\delta/3}$.

Picking the parameters - Take 2 ($\mathcal{S} = n^\delta$)

$O(r/s)$ regions $O\left(\frac{ns}{r}\right)$ edges / region $O\left(\frac{n\sqrt{s}}{r}\right)$ edges / boundary

- Need to set $r \leq \mathcal{S} = n^\delta$ in order to compute planar s-division (on one machine)
- Need to set $s \leq r$, choose $s = r^{2/3} = n^{2\delta/3}$.

$O(n^{\delta/3})$ regions $O\left(n^{1-\delta/3}\right)$ edges / region $O\left(n^{1-2\delta/3}\right)$ edges / boundary

Picking the parameters - Take 2 ($\mathcal{S} = n^\delta$)

$O(r/s)$ regions $O\left(\frac{ns}{r}\right)$ edges / region $O\left(\frac{n\sqrt{s}}{r}\right)$ edges / boundary

- Need to set $r \leq \mathcal{S} = n^\delta$ in order to compute planar s-division (on one machine)
- Need to set $s \leq r$, choose $s = r^{2/3} = n^{2\delta/3}$.

$O(n^{\delta/3})$ regions $O\left(n^{1-\delta/3}\right)$ edges / region $O\left(n^{1-2\delta/3}\right)$ edges / boundary

- **Recursively redraw** each region to get graph proportional to boundary size

Picking the parameters - Take 2 ($\mathcal{S} = n^\delta$)

$O(r/s)$ regions $\quad$ $O\left(\frac{ns}{r}\right)$ edges / region $\quad$ $O\left(\frac{n\sqrt{s}}{r}\right)$ edges / boundary

- Need to set $r \leq \mathcal{S} = n^\delta$ in order to compute planar s-division (on one machine)
- Need to set $s \leq r$, choose $s = r^{2/3} = n^{2\delta/3}$.

$O(n^{\delta/3})$ regions $\quad$ $O\left(n^{1-\delta/3}\right)$ edges / region $\quad$ $O\left(n^{1-2\delta/3}\right)$ edges / boundary

- **Recursively redraw** each region to get graph proportional to boundary size

Graph with $O(n)$ edges $\rightarrow$ Graph with $O(n^{1-\delta/3})$ edges.

Picking the parameters - Take 2 ($\mathcal{S} = n^\delta$)

$O(r/s)$ regions $O\left(\frac{ns}{r}\right)$ edges / region $O\left(\frac{n\sqrt{s}}{r}\right)$ edges / boundary

- Need to set $r \leq \mathcal{S} = n^\delta$ in order to compute planar s-division (on one machine)
- Need to set $s \leq r$, choose $s = r^{2/3} = n^{2\delta/3}$.

$O(n^{\delta/3})$ regions $O\left(n^{1-\delta/3}\right)$ edges / region $O\left(n^{1-2\delta/3}\right)$ edges / boundary

- **Recursively redraw** each region to get graph proportional to boundary size

Graph with $O(n)$ edges $\rightarrow$ Graph with $O(n^{1-\delta/3})$ edges.

Iterate this until graph is small!

Technical challenges

$O(n^{\delta/3})$ regions $O\left(n^{1-\delta/3}\right)$ edges / region $O\left(n^{1-2\delta/3}\right)$ edges / boundary

Technical Challenges

- The number of boundary edges is large $O(n^{1-2\delta/3})$

Technical challenges

$O(n^{\delta/3})$ regions $O\left(n^{1-\delta/3}\right)$ edges / region $O\left(n^{1-2\delta/3}\right)$ edges / boundary

Technical Challenges

- The number of boundary edges is large $O(n^{1-2\delta/3})$

Technical challenges

$O(n^{\delta/3})$ regions $\quad O\left(n^{1-\delta/3}\right)$ edges / region $\quad O\left(n^{1-2\delta/3}\right)$ edges / boundary

Technical Challenges

- The number of boundary edges is large $O(n^{1-2\delta/3})$
  - Solution: Return on multiple machines

Technical challenges

$O(n^{\delta/3})$ regions $\quad O\left(n^{1-\delta/3}\right)$ edges / region $\quad O\left(n^{1-2\delta/3}\right)$ edges / boundary

Technical Challenges

- The number of boundary edges is large $O(n^{1-2\delta/3})$
  - Solution: Return on multiple machines
- Base case?

Technical challenges

$O(n^{\delta/3})$ regions $\quad$ $O\left(n^{1-\delta/3}\right)$ edges / region $\quad$ $O\left(n^{1-2\delta/3}\right)$ edges / boundary

Technical Challenges

- The number of boundary edges is large $O(n^{1-2\delta/3})$
  - Solution: Return on multiple machines
- Base case?
  - Solution: Graph drawing!

Technical challenges

$O(n^{\delta/3})$ regions $\quad O\left(n^{1-\delta/3}\right)$ edges / region $\quad O\left(n^{1-2\delta/3}\right)$ edges / boundary

Technical Challenges

- The number of boundary edges is large $O(n^{1-2\delta/3})$
  - Solution: Return on multiple machines
- Base case?
  - Solution: Graph drawing!
- Precision issues?

Technical issues (cont.)

$O(n^{\delta/3})$ regions $O\left(n^{1-\delta/3}\right)$ edges / region $O\left(n^{1-2\delta/3}\right)$ edges / boundary

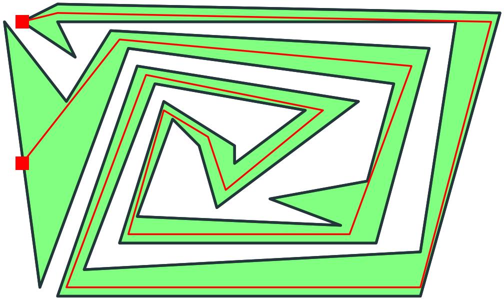
The boundary of a region is a complex polygon of size $O(n^{\delta/3})$ with $O(1)$ holes

Technical issues (cont.)

$O(n^{\delta/3})$ regions $\quad$ $O\left(n^{1-\delta/3}\right)$ edges / region $\quad$ $O\left(n^{1-2\delta/3}\right)$ edges / boundary

The boundary of a region is a complex polygon of size $O(n^{\delta/3})$ with $O(1)$ holes

- Connecting two points may need $O(n^{\delta/3})$ bends VERY BAD!



Technical issues (cont.)

$O(n^{\delta/3})$ regions $\quad O\left(n^{1-\delta/3}\right)$ edges / region $\quad O\left(n^{1-2\delta/3}\right)$ edges / boundary

The boundary of a region is a complex polygon of size $O(n^{\delta/3})$ with $O(1)$ holes

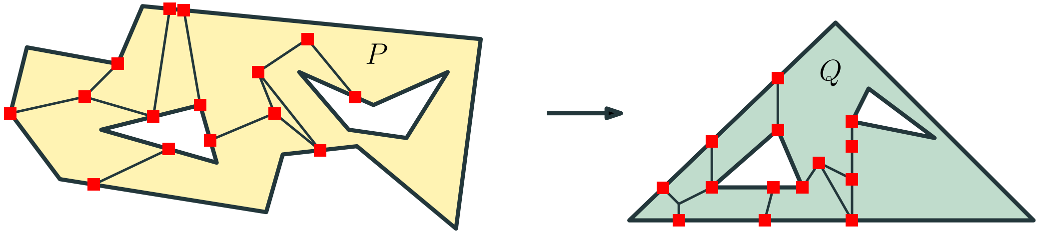
- Connecting two points may need $O(n^{\delta/3})$ bends VERY BAD!
- Solution: Draw subproblem in a triangle w/ triangular holes

Technical issues (cont.)

$O(n^{\delta/3})$ regions $\quad$ $O\left(n^{1-\delta/3}\right)$ edges / region $\quad$ $O\left(n^{1-2\delta/3}\right)$ edges / boundary

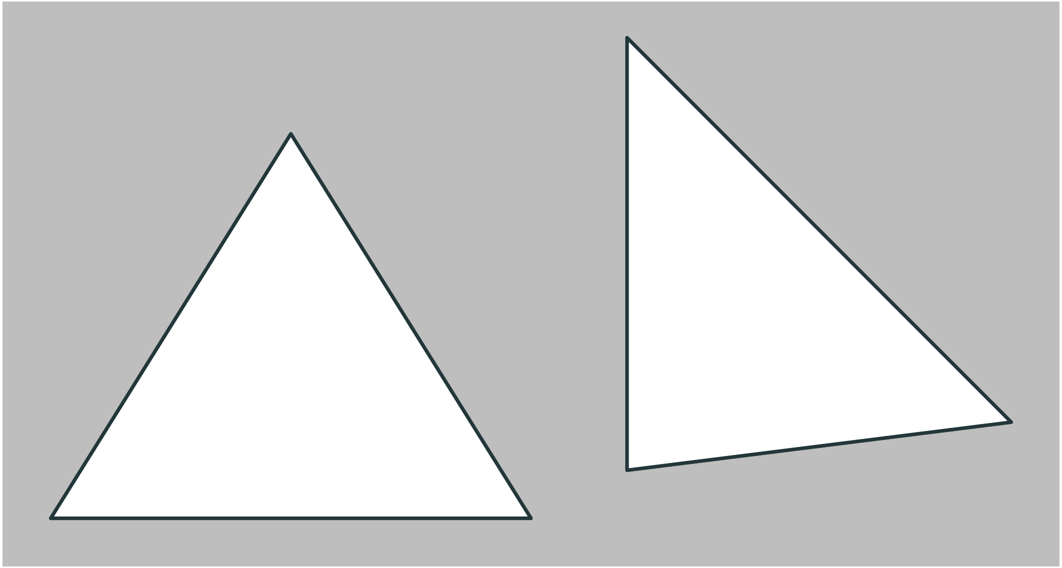
The boundary of a region is a complex polygon of size $O(n^{\delta/3})$ with $O(1)$ holes

- Connecting two points may need $O(n^{\delta/3})$ bends VERY BAD!
- Solution: Draw subproblem in a triangle w/ triangular holes

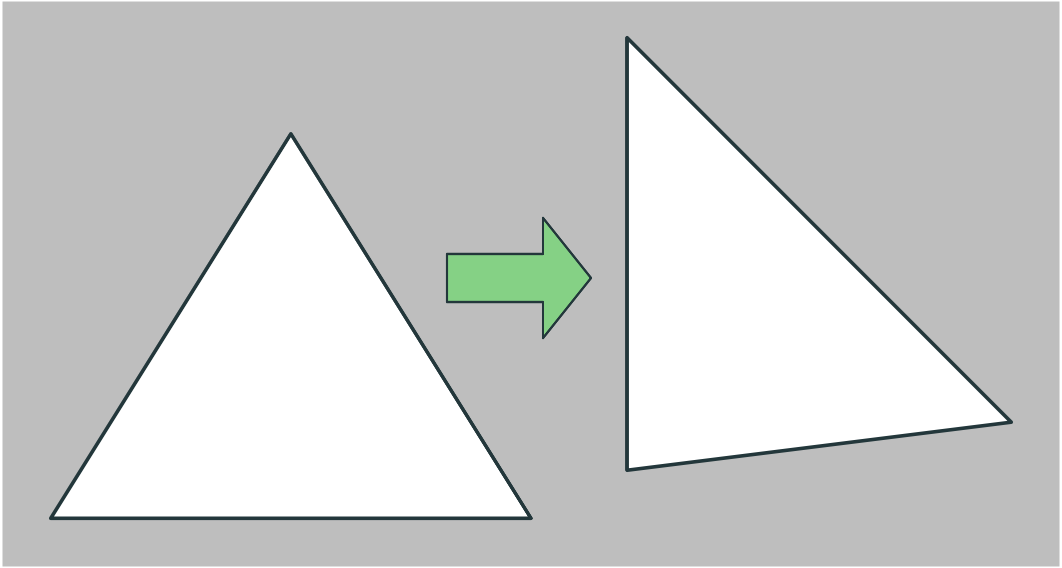


Advantages of triangular boundaries

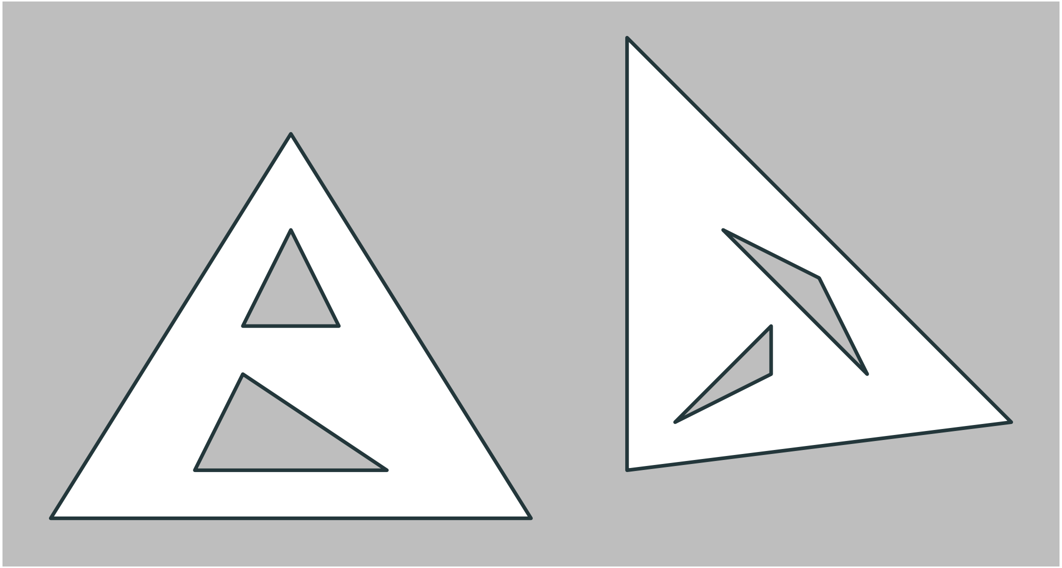
Compatible triangulations



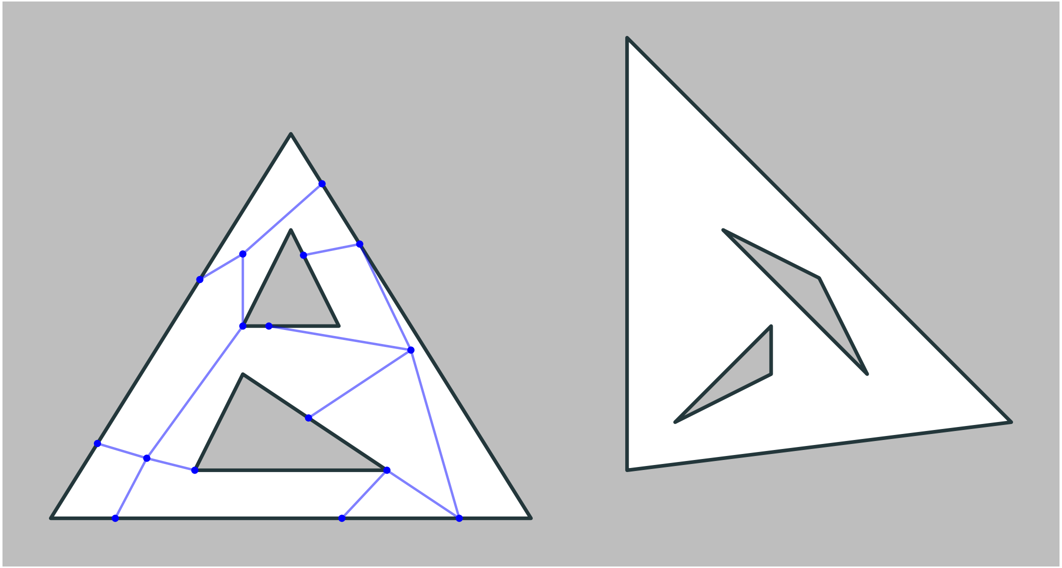
Compatible triangulations



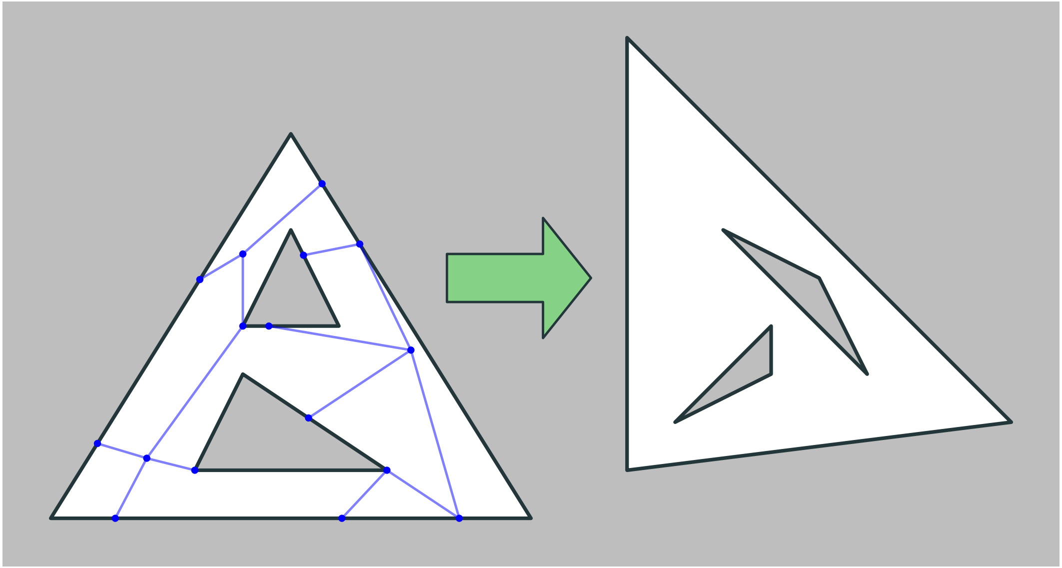
Compatible triangulations



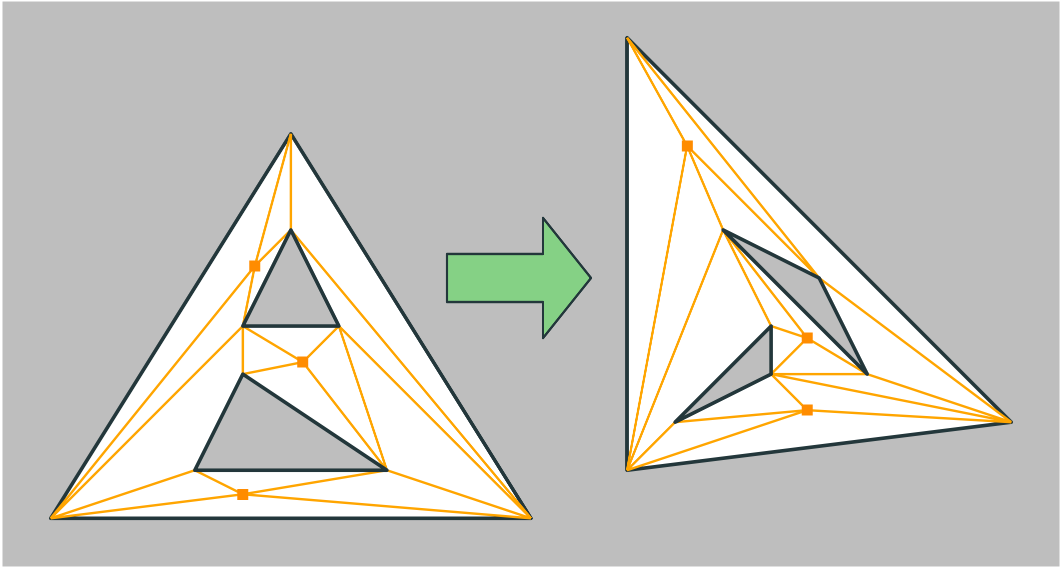
Compatible triangulations



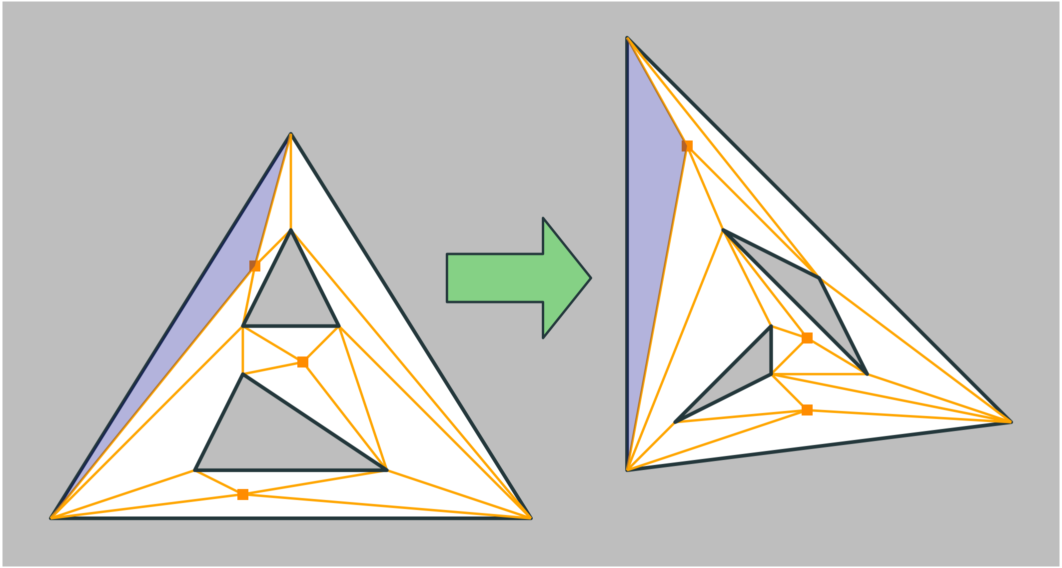
Compatible triangulations



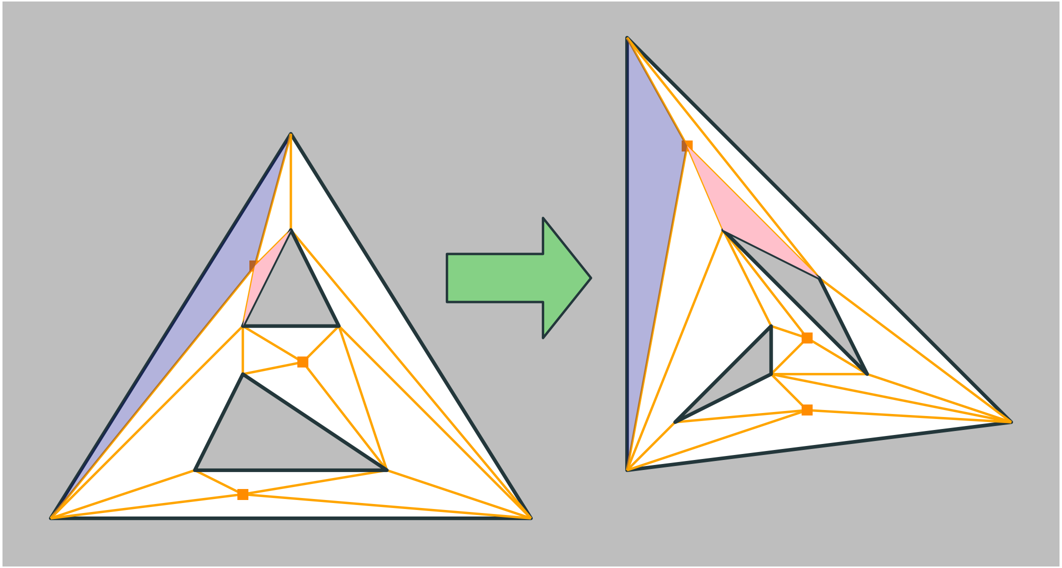
Compatible triangulations



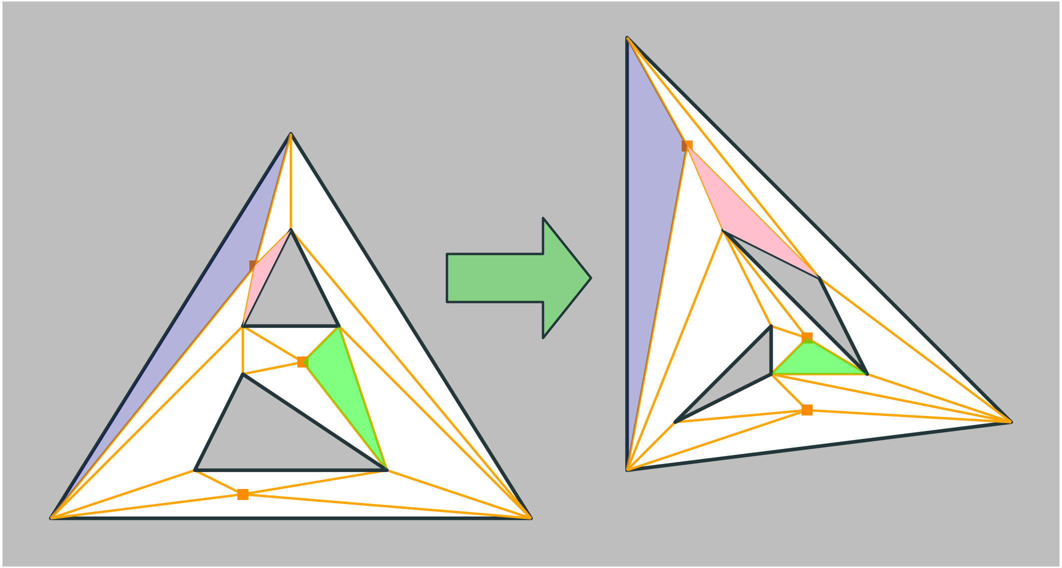
Compatible triangulations



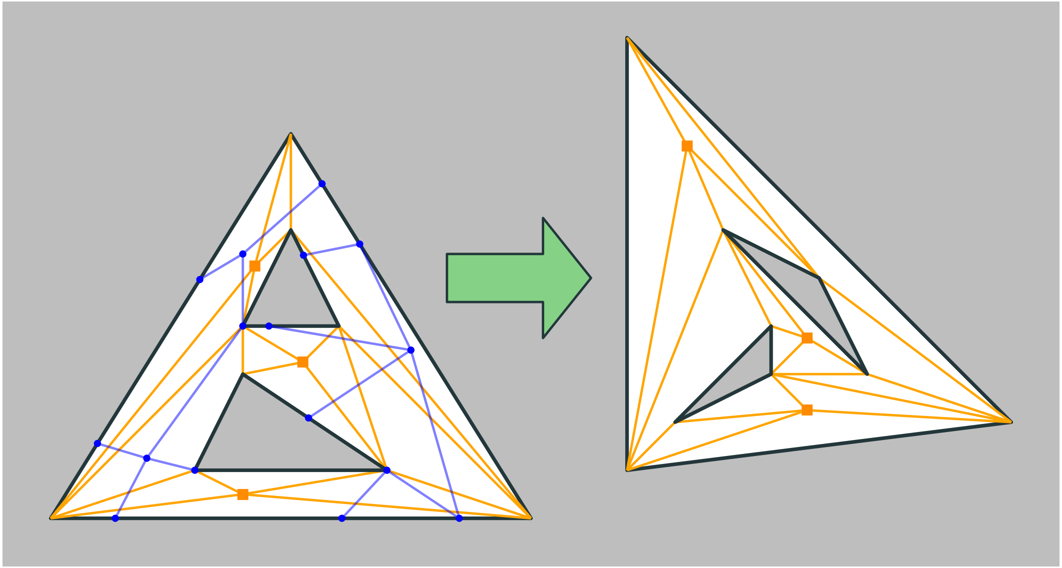
Compatible triangulations



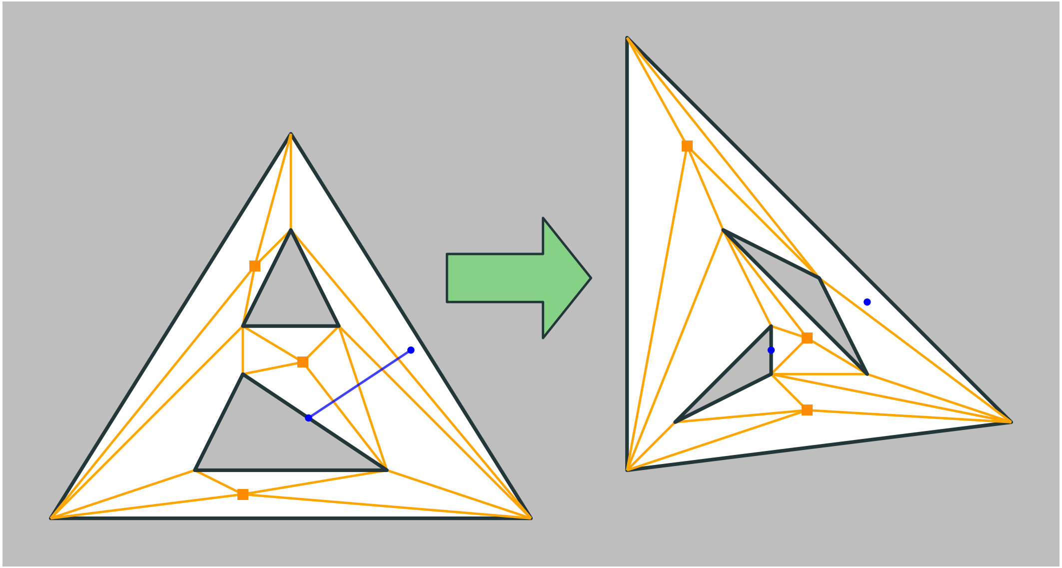
Compatible triangulations



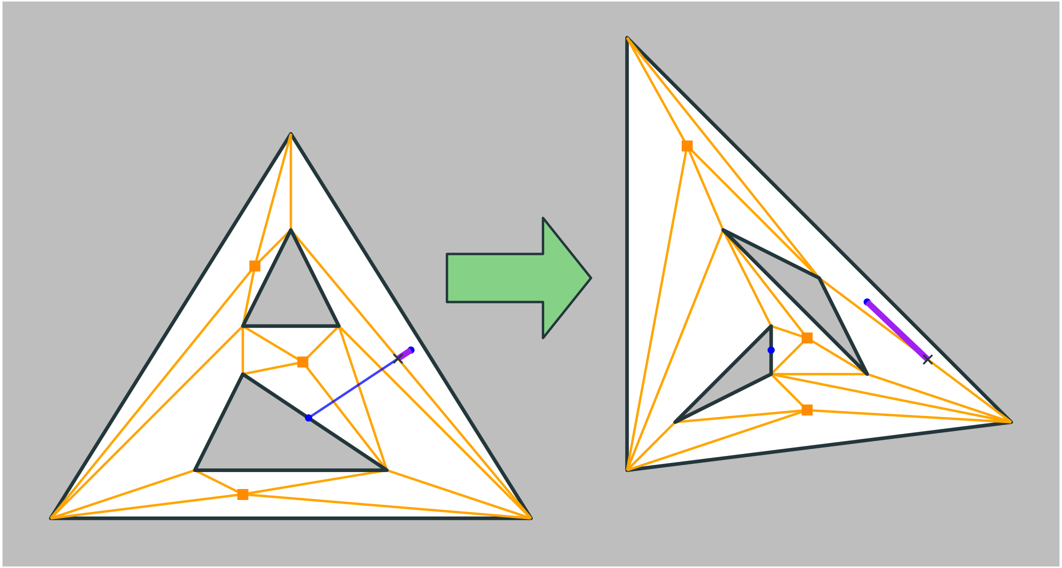
Compatible triangulations



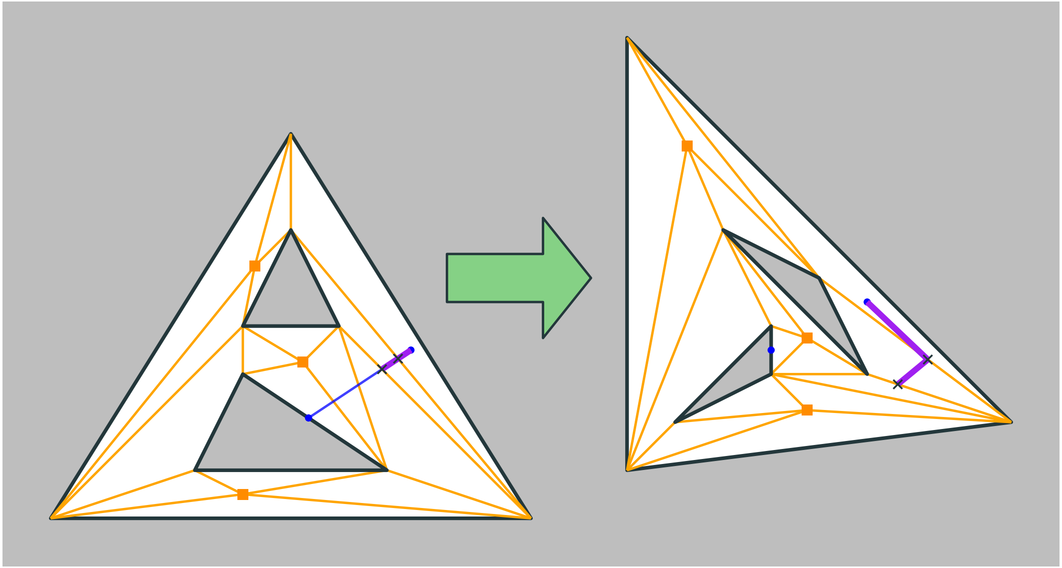
Compatible triangulations



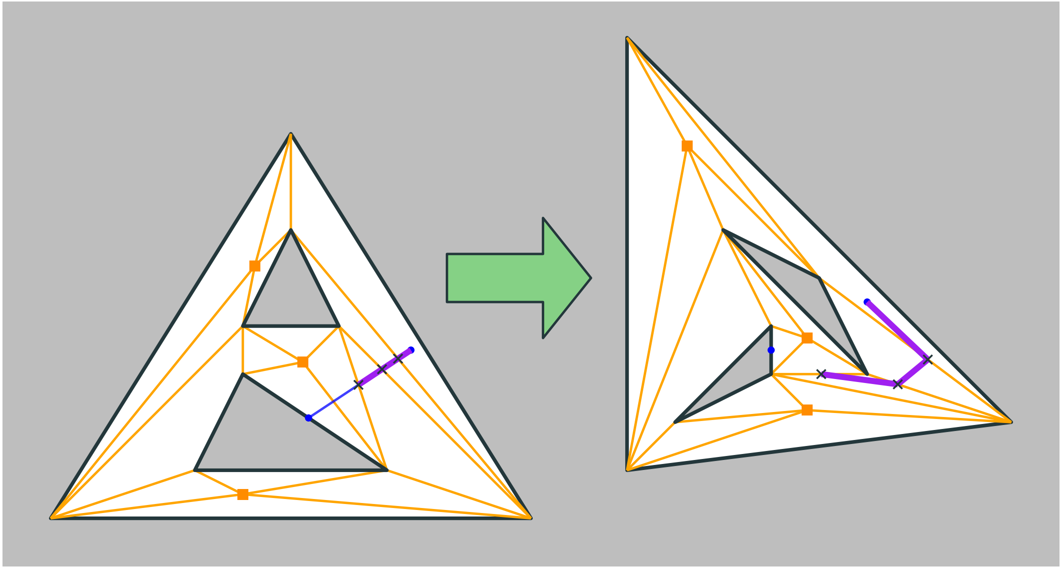
Compatible triangulations



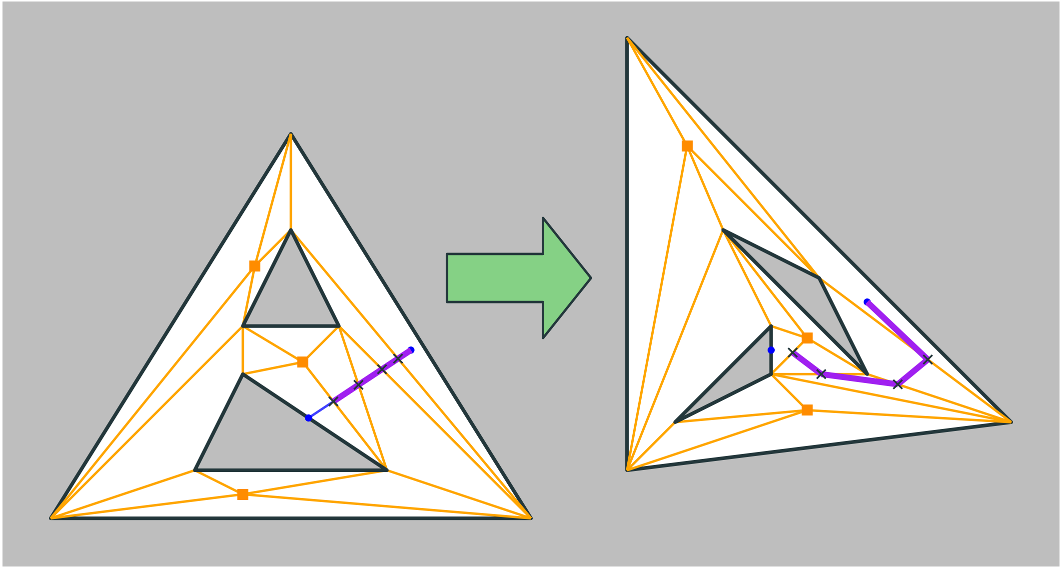
Compatible triangulations



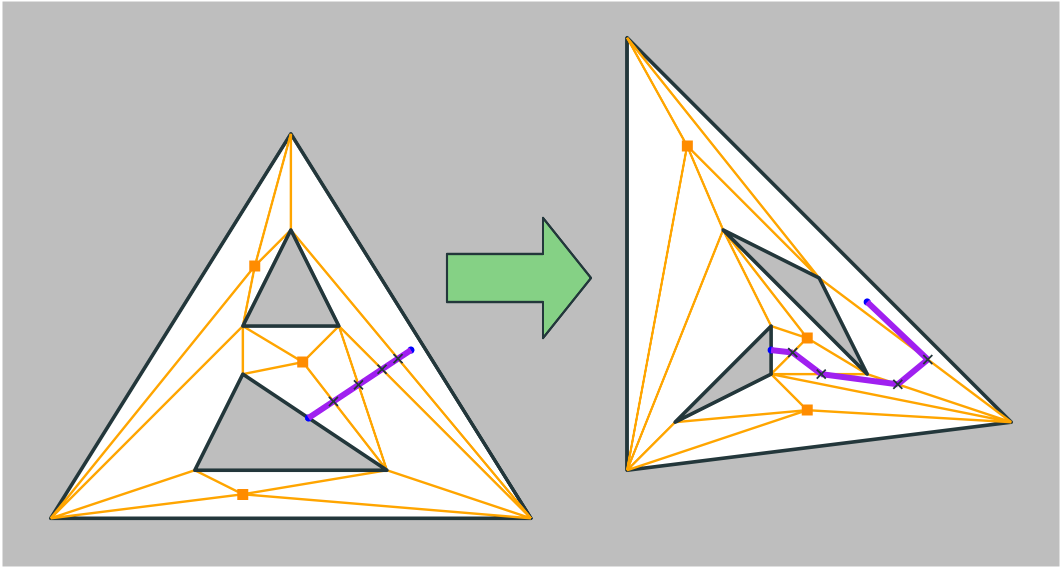
Compatible triangulations



Compatible triangulations

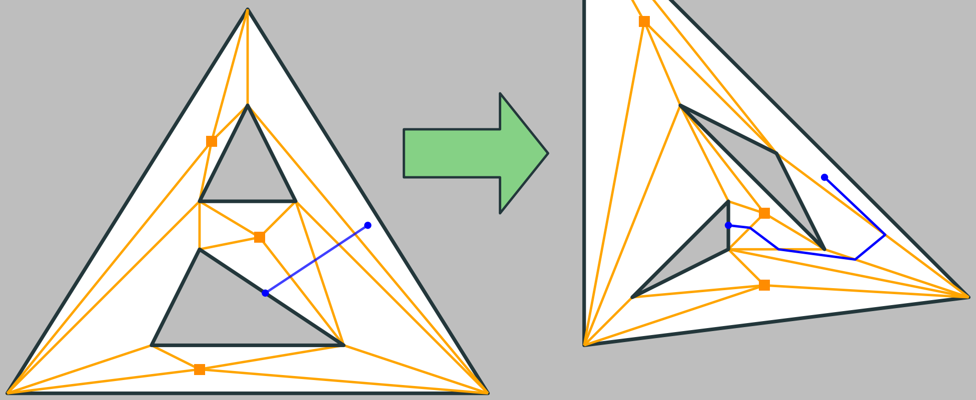


Compatible triangulations



Compatible triangulations

h holes $\Rightarrow O(h^2)$ bends



Technical issues in the gluing step (cont.)

We need to join these triangles together.

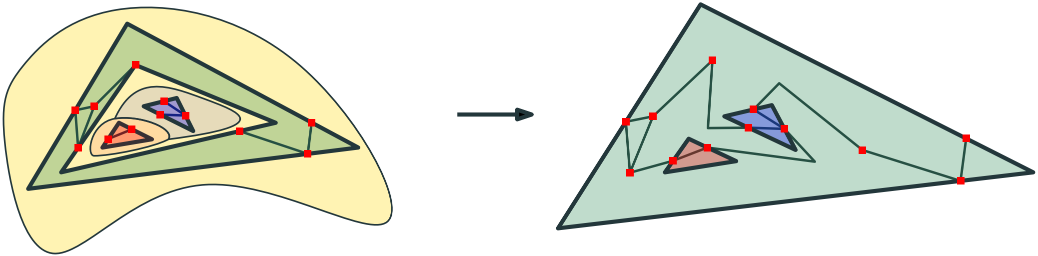
Subdivision may have nested holes (can be nested $O(n^{\delta/3})$ deep).

Technical issues in the gluing step (cont.)

We need to join these triangles together.

Subdivision may have nested holes (can be nested $O(n^{\delta/3})$ deep).

- Need to join boundaries of adjacent subproblems together **in parallel**.

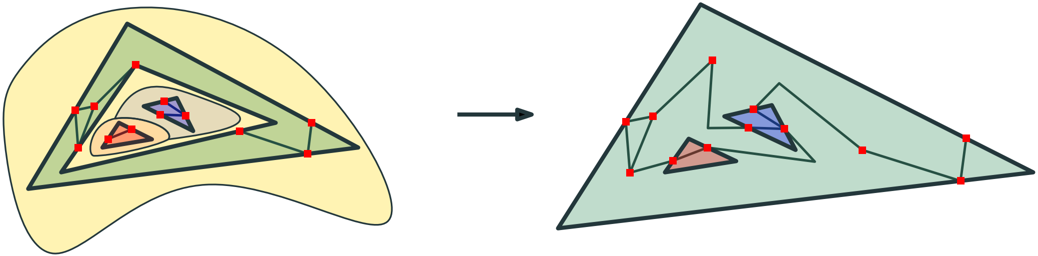


Technical issues in the gluing step (cont.)

We need to join these triangles together.

Subdivision may have nested holes (can be nested $O(n^{\delta/3})$ deep).

- Need to join boundaries of adjacent subproblems together **in parallel**.

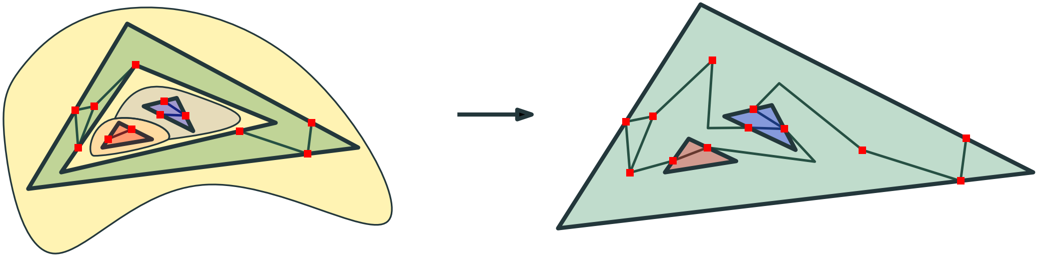


Technical issues in the gluing step (cont.)

We need to join these triangles together.

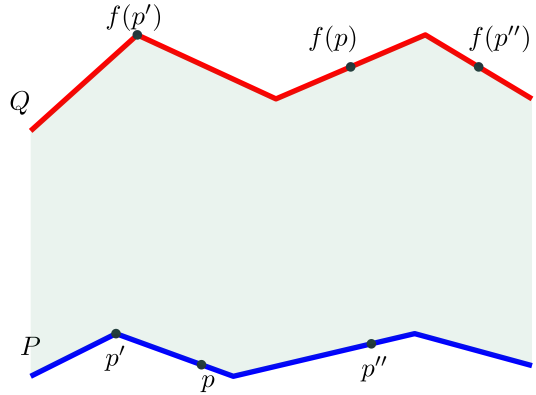
Subdivision may have nested holes (can be nested $O(n^{\delta/3})$ deep).

- Need to join boundaries of adjacent subproblems together **in parallel**.
- Solution: Parallel routing technique + a **scaffold graph**.

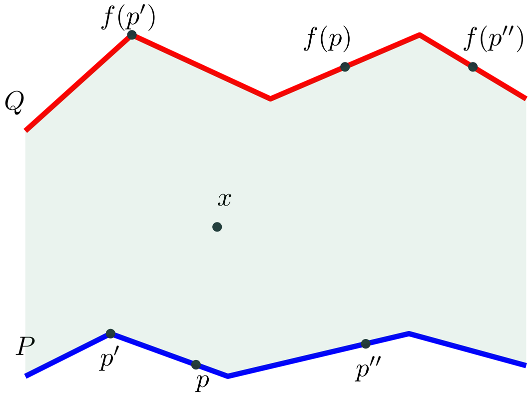


Routing between curves in parallel

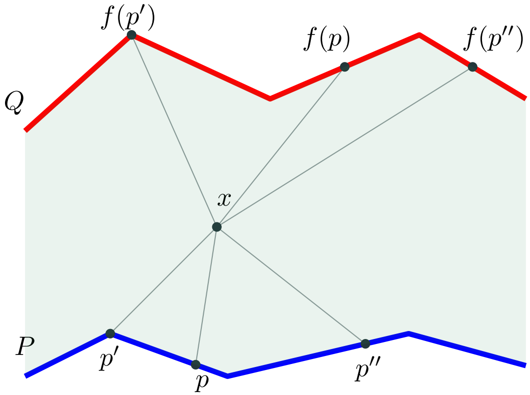
Routing between curves in parallel



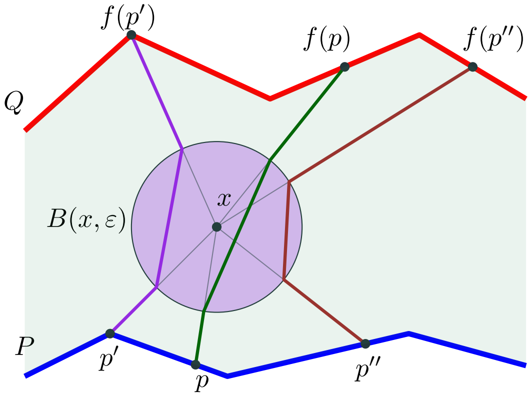
Routing between curves in parallel



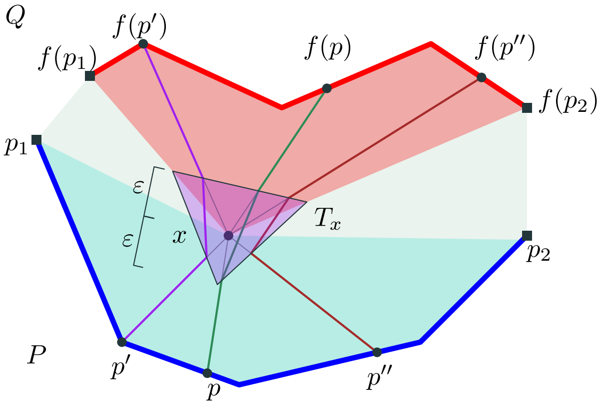
Routing between curves in parallel



Routing between curves in parallel

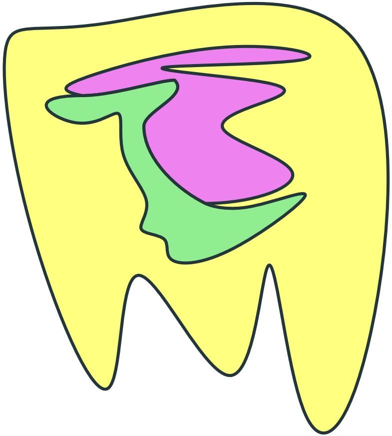


Routing between curves in parallel

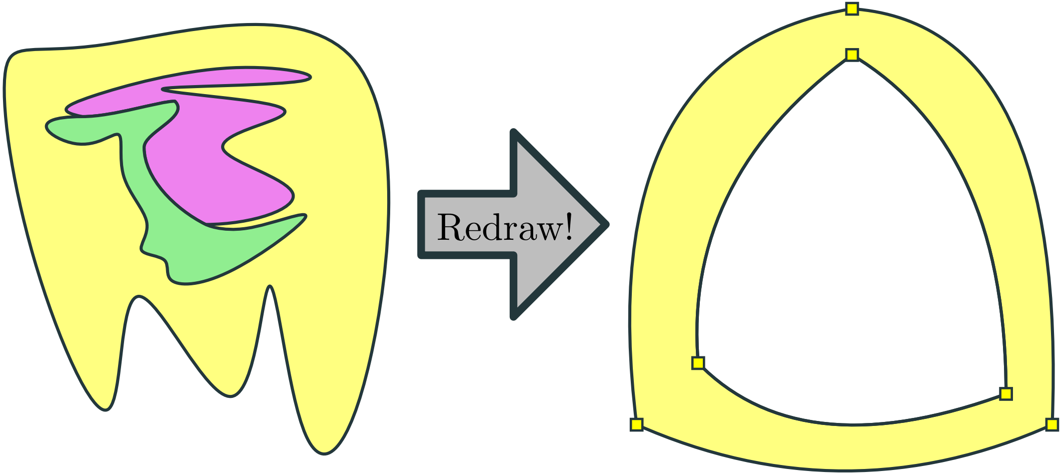


Scaffold construction

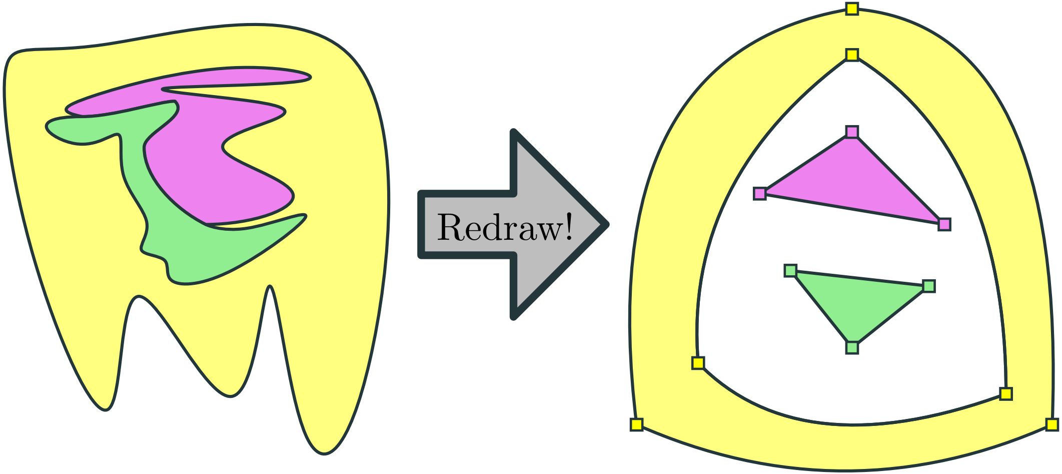
Scaffold construction (First attempt)



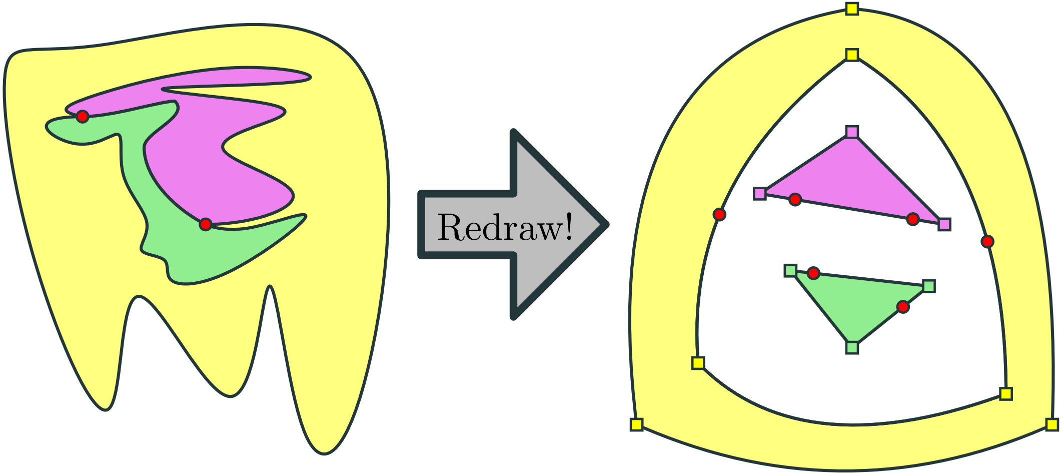
Scaffold construction (First attempt)



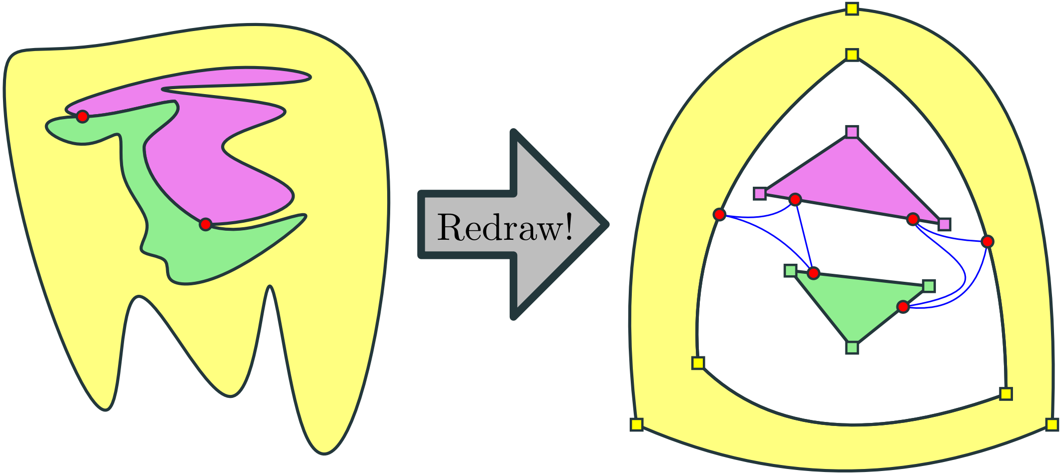
Scaffold construction (First attempt)



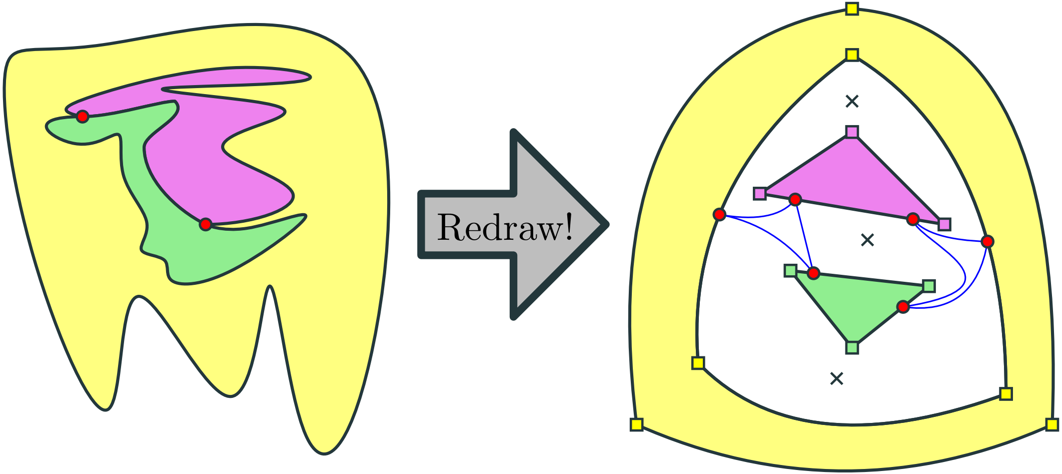
Scaffold construction (First attempt)



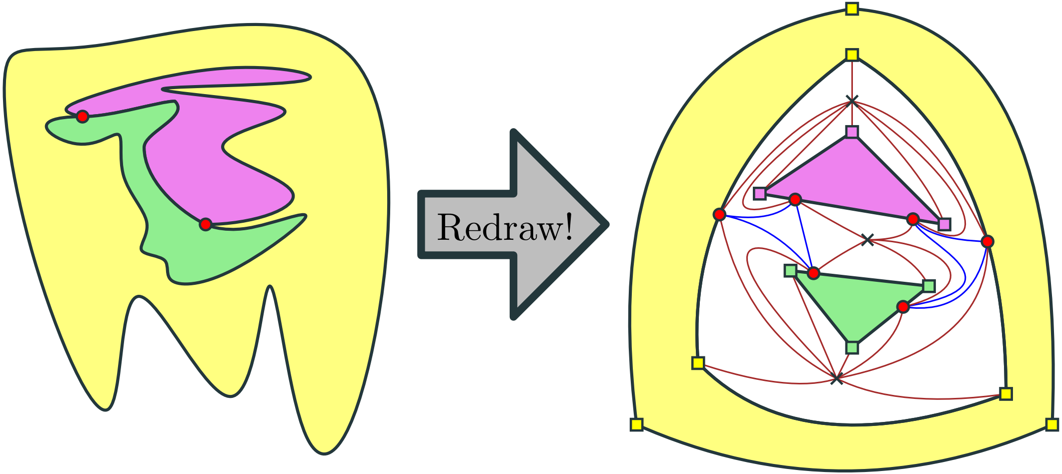
Scaffold construction (First attempt)



Scaffold construction (First attempt)

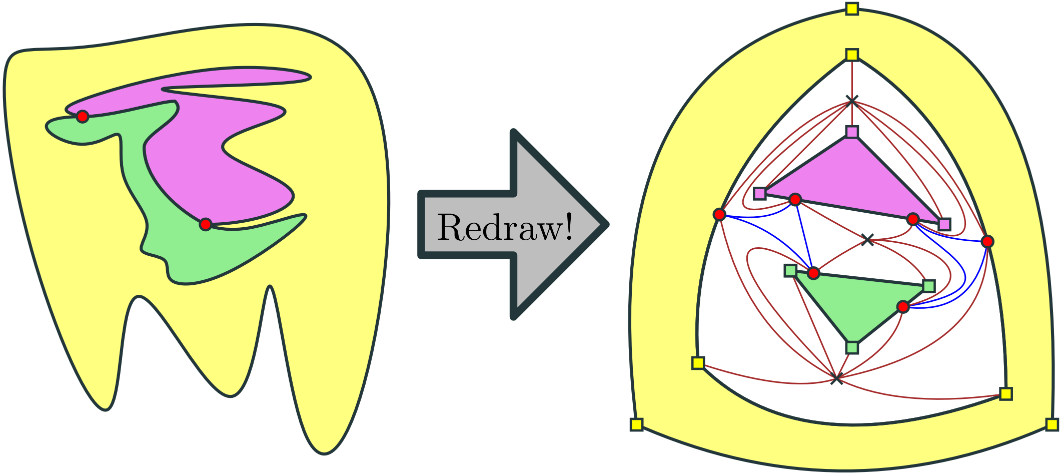


Scaffold construction (First attempt)



Scaffold construction (First attempt)

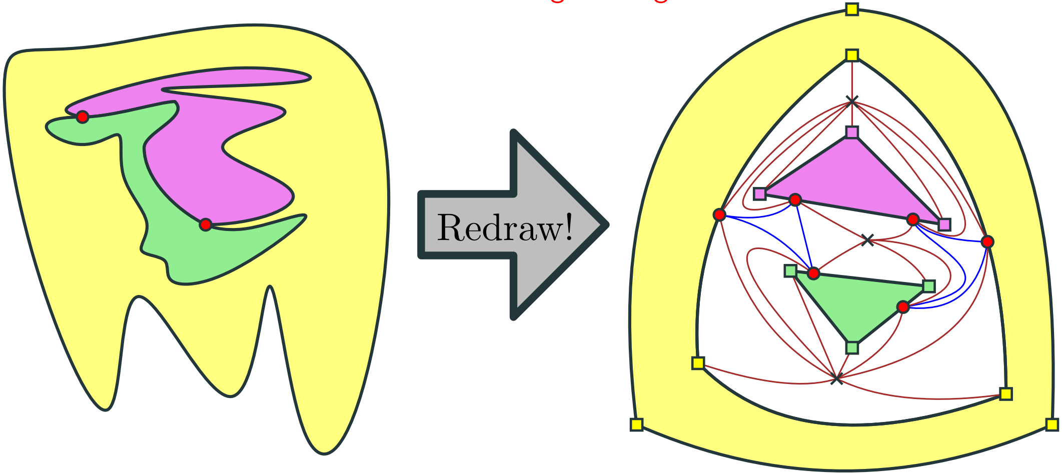
The scaffold has size $O(n^{\delta/3}) < \mathcal{S}$ so can be stored (and drawn) by all machines!



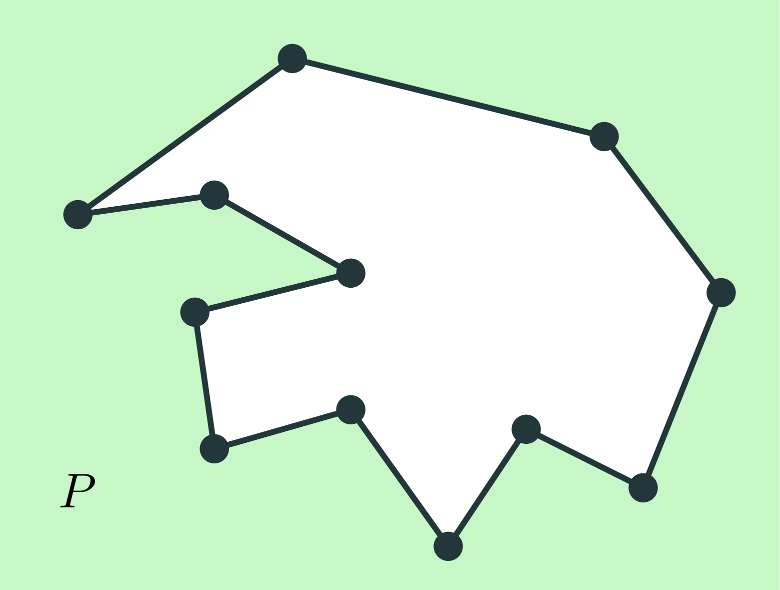
Scaffold construction (First attempt)

The scaffold has size $O(n^{\delta/3}) < \mathcal{S}$ so can be stored (and drawn) by all machines!

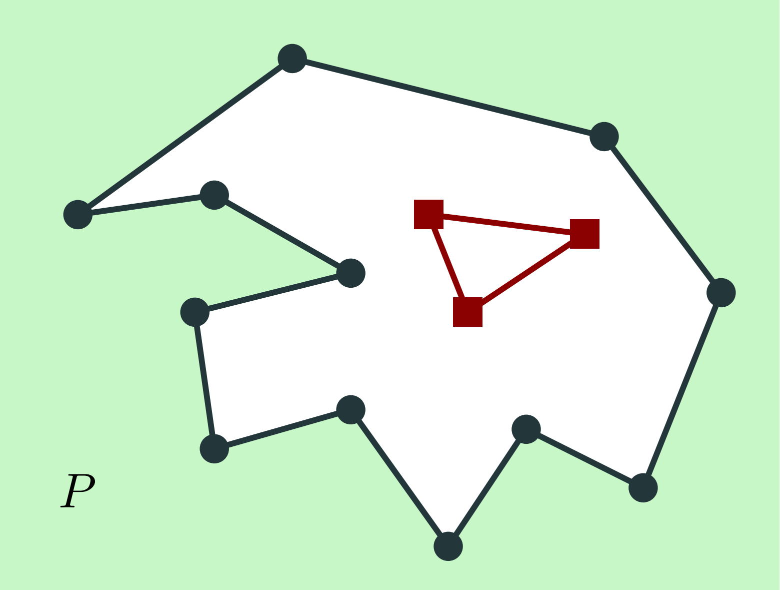
Faces are no longer triangles!



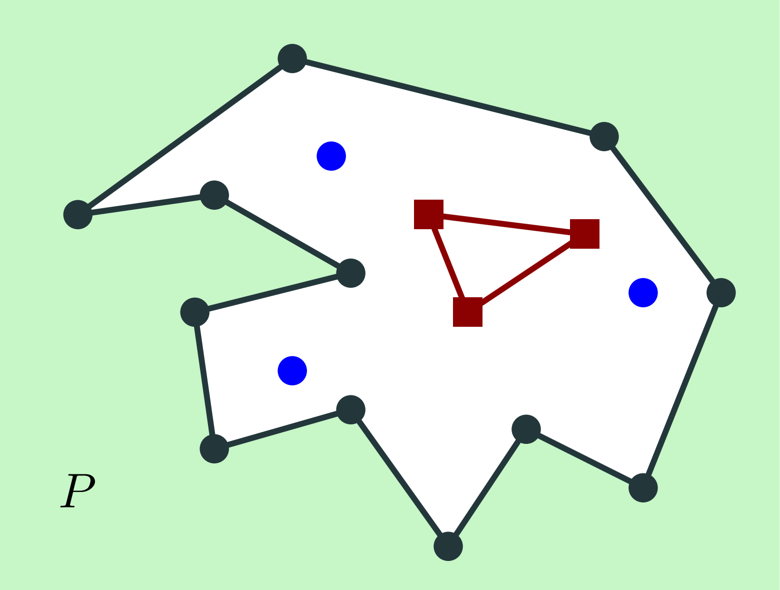
Scaffold construction for a polygon



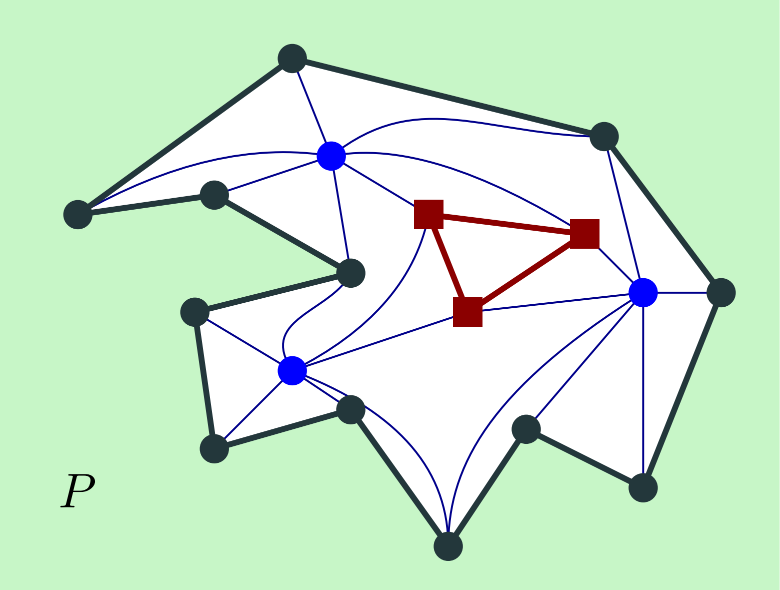
Scaffold construction for a polygon



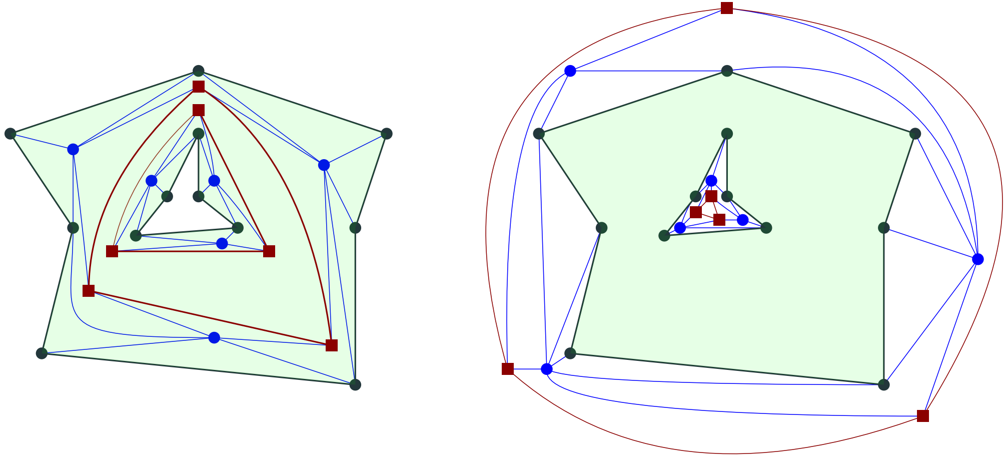
Scaffold construction for a polygon



Scaffold construction for a polygon



Inside and outside scaffolding



Application to problems

Application to problems

- **Connected components** - keep track of components, replace with tree

Application to problems

- **Connected components** - keep track of components, replace with tree
- **MST** - replace with MST of subgraph (contract non-boundary)

Application to problems

- **Connected components** - keep track of components, replace with tree
- **MST** - replace with MST of subgraph (contract non-boundary)
- **Euclidean MST** - MST in Delaunay triangulation ($O(1)$ rounds by [ANOY14])

Application to problems

- **Connected components** - keep track of components, replace with tree
- **MST** - replace with MST of subgraph (contract non-boundary)
- **Euclidean MST** - MST in Delaunay triangulation ($O(1)$ rounds by [ANOY14])
- $(1+\varepsilon)$**-approx SSSP** - replace with ε-emulator of Chang et al. [CKT22].

Application to problems

- **Connected components** - keep track of components, replace with tree
- **MST** - replace with MST of subgraph (contract non-boundary)
- **Euclidean MST** - MST in Delaunay triangulation ($O(1)$ rounds by [ANOY14])
- $(1+\varepsilon)$-**approx SSSP** - replace with ε-emulator of Chang et al. [CKT22].
- $(1+\varepsilon)$-**approx APSP** - Similar to SSSP

Application to problems

- **Connected components** - keep track of components, replace with tree
- **MST** - replace with MST of subgraph (contract non-boundary)
- **Euclidean MST** - MST in Delaunay triangulation ($O(1)$ rounds by [ANOY14])
- $(1+\varepsilon)$**-approx SSSP** - replace with ε-emulator of Chang et al. [CKT22].
- $(1+\varepsilon)$**-approx APSP** - Similar to SSSP
- $(1+\varepsilon)$**-approx weighted edit distance** - DP is SSSP in a grid-like graph

Application to problems

- **Connected components** - keep track of components, replace with tree
- **MST** - replace with MST of subgraph (contract non-boundary)
- **Euclidean MST** - MST in Delaunay triangulation ($O(1)$ rounds by [ANOY14])
- $(1+\varepsilon)$**-approx SSSP** - replace with ε-emulator of Chang et al. [CKT22].
- $(1+\varepsilon)$**-approx APSP** - Similar to SSSP
- $(1+\varepsilon)$**-approx weighted edit distance** - DP is SSSP in a grid-like graph
- $(1+\varepsilon)$**-approx shortest cycle** - recursive solution + recurse on ε-emulators

Application to problems

- **Connected components** - keep track of components, replace with tree
- **MST** - replace with MST of subgraph (contract non-boundary)
- **Euclidean MST** - MST in Delaunay triangulation ($O(1)$ rounds by [ANOY14])
- $(1+\varepsilon)$**-approx SSSP** - replace with ε-emulator of Chang et al. [CKT22].
- $(1+\varepsilon)$**-approx APSP** - Similar to SSSP
- $(1+\varepsilon)$**-approx weighted edit distance** - DP is SSSP in a grid-like graph
- $(1+\varepsilon)$**-approx shortest cycle** - recursive solution + recurse on ε-emulators
- $(1+\varepsilon)$**-approx flow/cut** - shortest cycle + planar graph duality

Application to problems

- **Connected components** - keep track of components, replace with tree
- **MST** - replace with MST of subgraph (contract non-boundary)
- **Euclidean MST** - MST in Delaunay triangulation ($O(1)$ rounds by [ANOY14])
- $(1+\varepsilon)$**-approx SSSP** - replace with ε-emulator of Chang et al. [CKT22].
- $(1+\varepsilon)$**-approx APSP** - Similar to SSSP
- $(1+\varepsilon)$**-approx weighted edit distance** - DP is SSSP in a grid-like graph
- $(1+\varepsilon)$**-approx shortest cycle** - recursive solution + recurse on ε-emulators
- $(1+\varepsilon)$**-approx flow/cut** - shortest cycle + planar graph duality

Conclusion

Open questions in the fully scalable regime

- EMST in higher dimensions?

Conclusion

Open questions in the fully scalable regime

- EMST in higher dimensions? ($O(1)$-approx in $O(\log \log n)$ [JMNZ23]

Conclusion

Open questions in the fully scalable regime

- EMST in higher dimensions? ($O(1)$-approx in $O(\log \log n)$ [JMNZ23]
- Approximate diameter and radius?

Conclusion

Open questions in the fully scalable regime

- EMST in higher dimensions? ($O(1)$-approx in $O(\log \log n)$ [JMNZ23]
- Approximate diameter and radius?
- Exact solutions to distance-based problems?

Conclusion

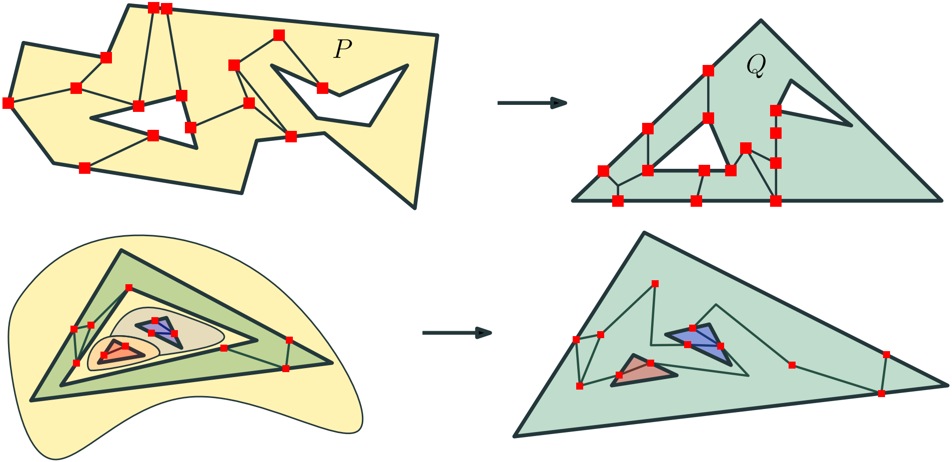
Open questions in the fully scalable regime

- EMST in higher dimensions? ($O(1)$-approx in $O(\log \log n)$ [JMNZ23]
- Approximate diameter and radius?
- Exact solutions to distance-based problems?
- Geometric intersection graphs?

Conclusion

Open questions in the fully scalable regime

- EMST in higher dimensions? ($O(1)$-approx in $O(\log \log n)$ [JMNZ23]
- Approximate diameter and radius?
- Exact solutions to distance-based problems?
- Geometric intersection graphs?
- Planarity testing and graph drawing?


P
Q

References

Alexandr Andoni, Aleksandar Nikolov, Krzysztof Onak, and Grigory Yaroslavtsev.
Parallel algorithms for geometric graph problems.
In David B. Shmoys, editor, Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014, pages 574–583. ACM, 2014.

Mahdi Boroujeni, Mohammad Ghodsi, and Saeed Seddighin.
Improved MPC algorithms for edit distance and Ulam distance.
IEEE Trans. Parallel Distributed Syst., 32(11):2764–2776, 2021.

Hsien-Chih Chang, Robert Krauthgamer, and Zihan Tan.
Almost-linear ε-emulators for planar graphs.
In Stefano Leonardi and Anupam Gupta, editors, STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022, pages 1311–1324. ACM, 2022.

MohammadTaghi Hajiaghayi, Saeed Seddighin, and Xiaorui Sun.
Massively parallel approximation algorithms for edit distance and longest common subsequence.
In Timothy M. Chan, editor, Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019, pages 1654–1672. SIAM, 2019.

Jacob Holm and Jakub Tětek.
Massively parallel computation on embedded planar graphs.
In Nikhil Bansal and Viswanath Nagarajan, editors, Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023, pages 4373–4408. SIAM, 2023.

Rajesh Jayaram, Vahab Mirrokni, Shyam Narayanan, and Peilin Zhong.
Massively parallel algorithms for high-dimensional euclidean minimum spanning tree.
CoRR, abs/2308.00503, 2023.