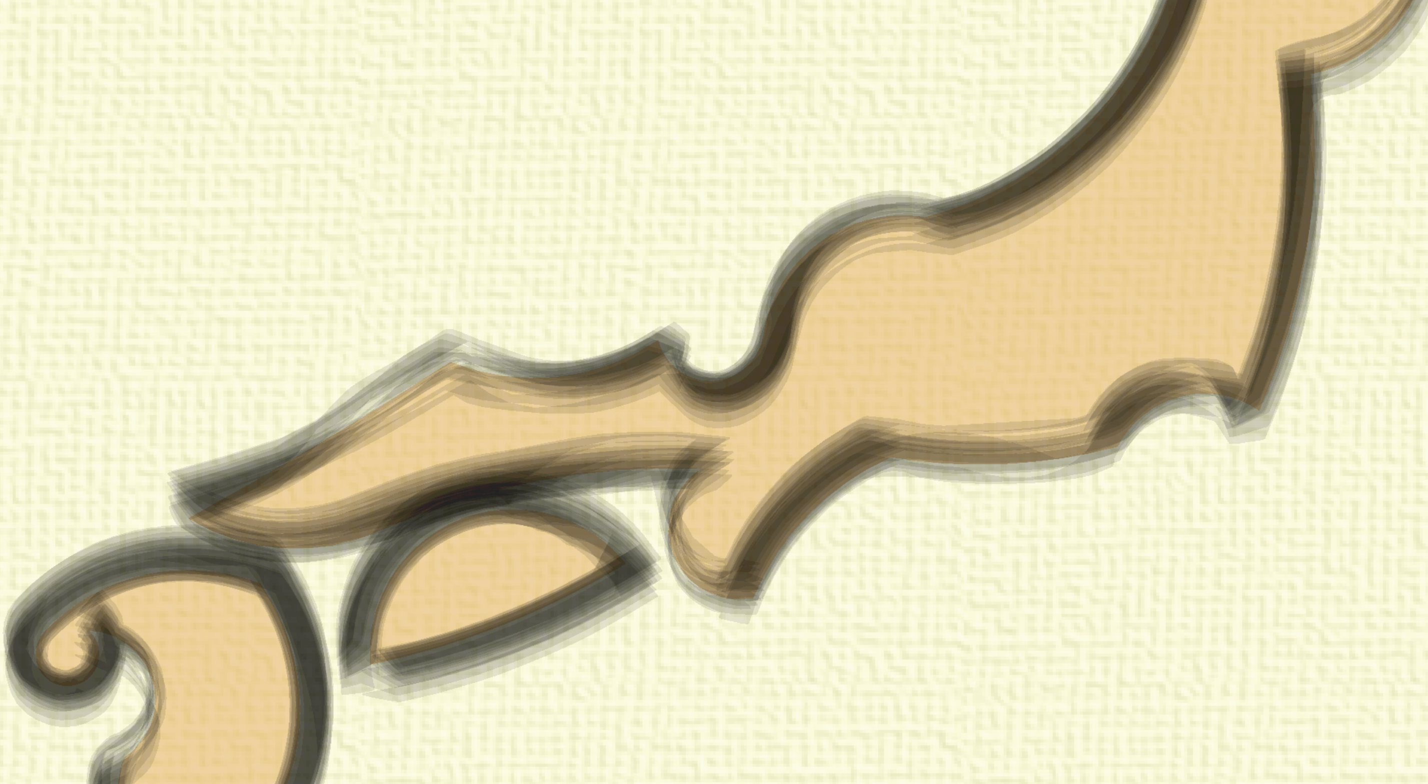


Schematization with and without geography

Wouter Meulemans

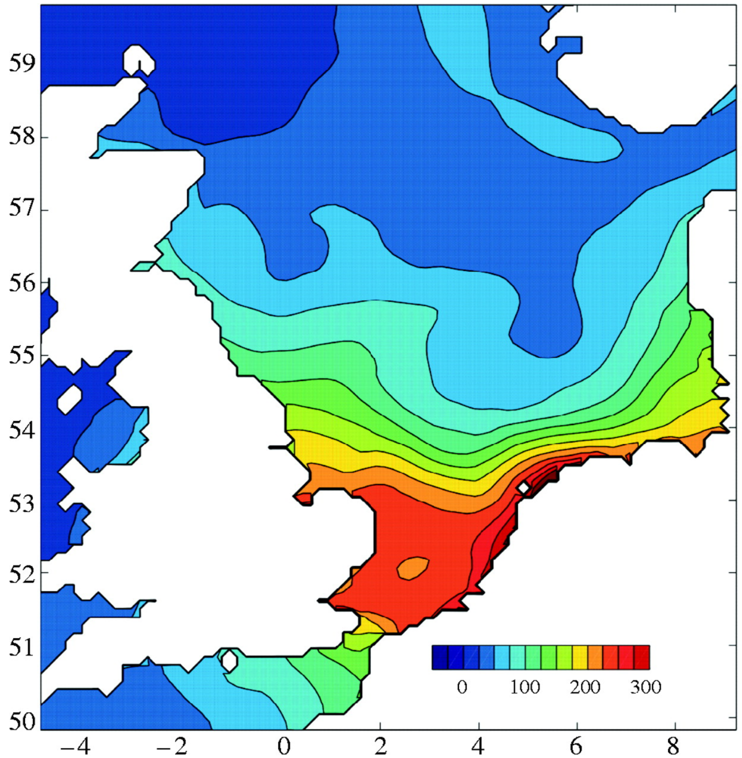
CITY UNIVERSITY LONDON
EST 1894

Middlesex University
November 17th, 2015

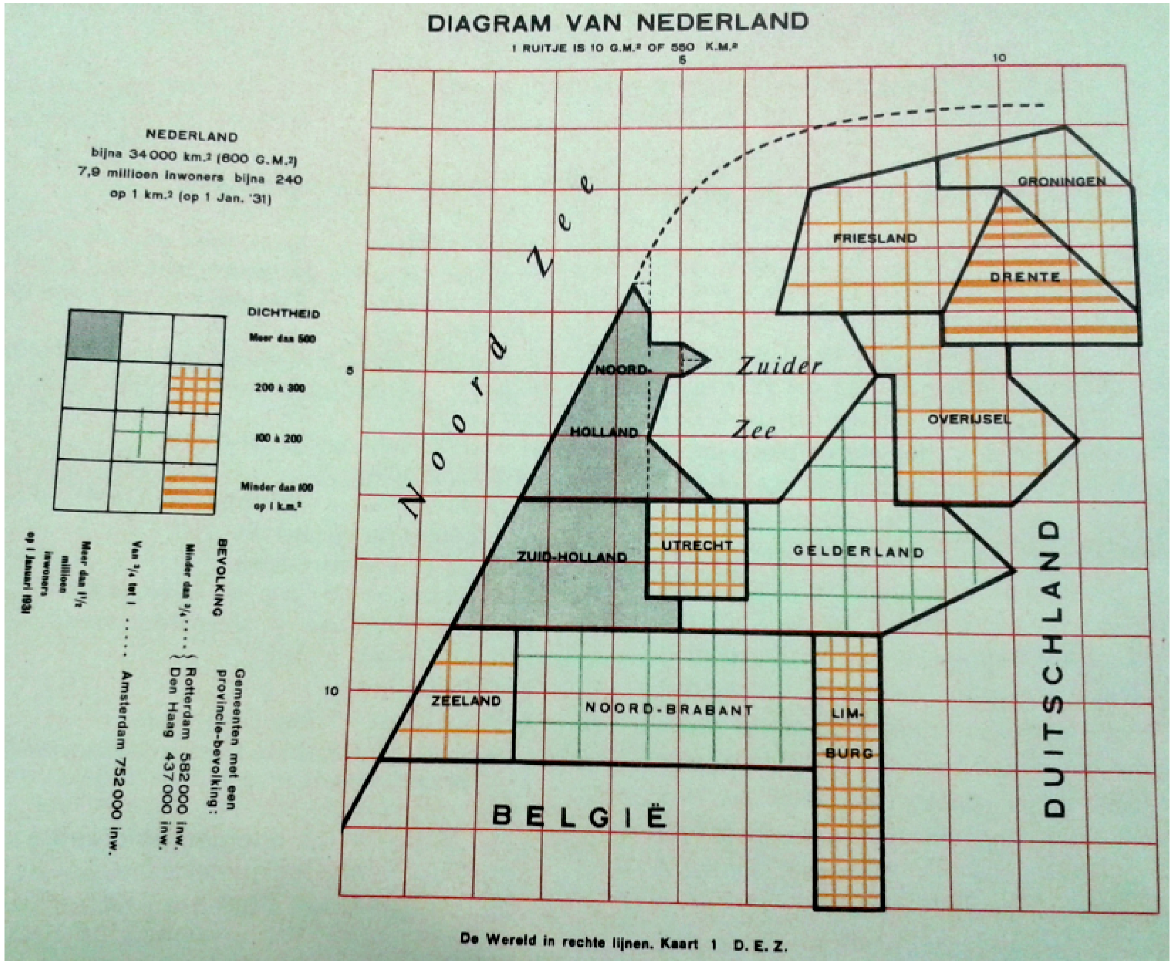
A schematic map

[http://www.nationalrail.co.uk]

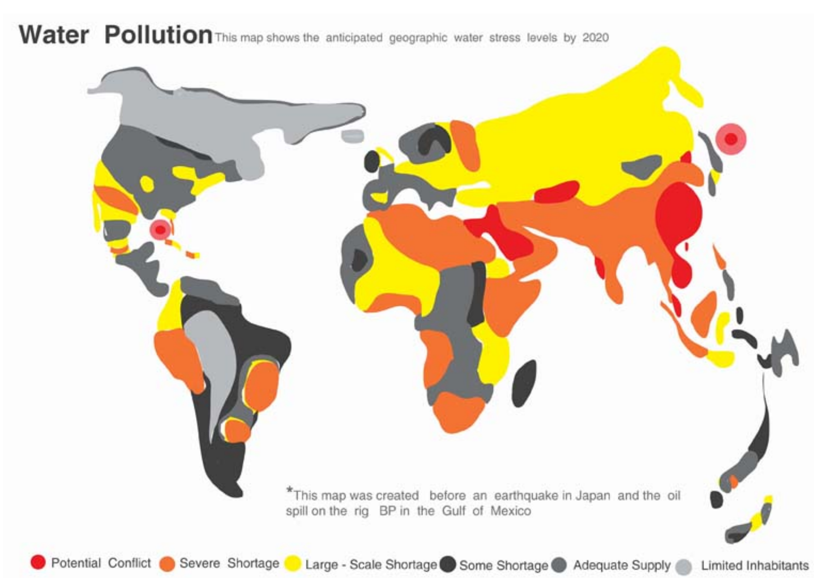
More than transit maps!



[Wolf & Flather, 2005]



[Zuidhof, 1932]



[http://ftrctlb.com/node/169]

Methods

	Networks	Regions
Lines	[Cabello et al, 2005] [Merrick & Gudmundsson, 2007] [Nöllenburg & Wolff, 2010]	[Buchin et al, 2011] [Cicerone & Cermignani, 2012] [Buchin et al, to appear]
Bézier	[Fink et al, 2013]	[Van Goethem et al, 2013]
Circular arcs	[Fink et al, 2014] [Van Goethem et al, 2014]	[Drysdale et al, 2008] [Heimlich & Held, 2008] [Van Goethem et al, 2013] [Van Goethem et al, 2015]

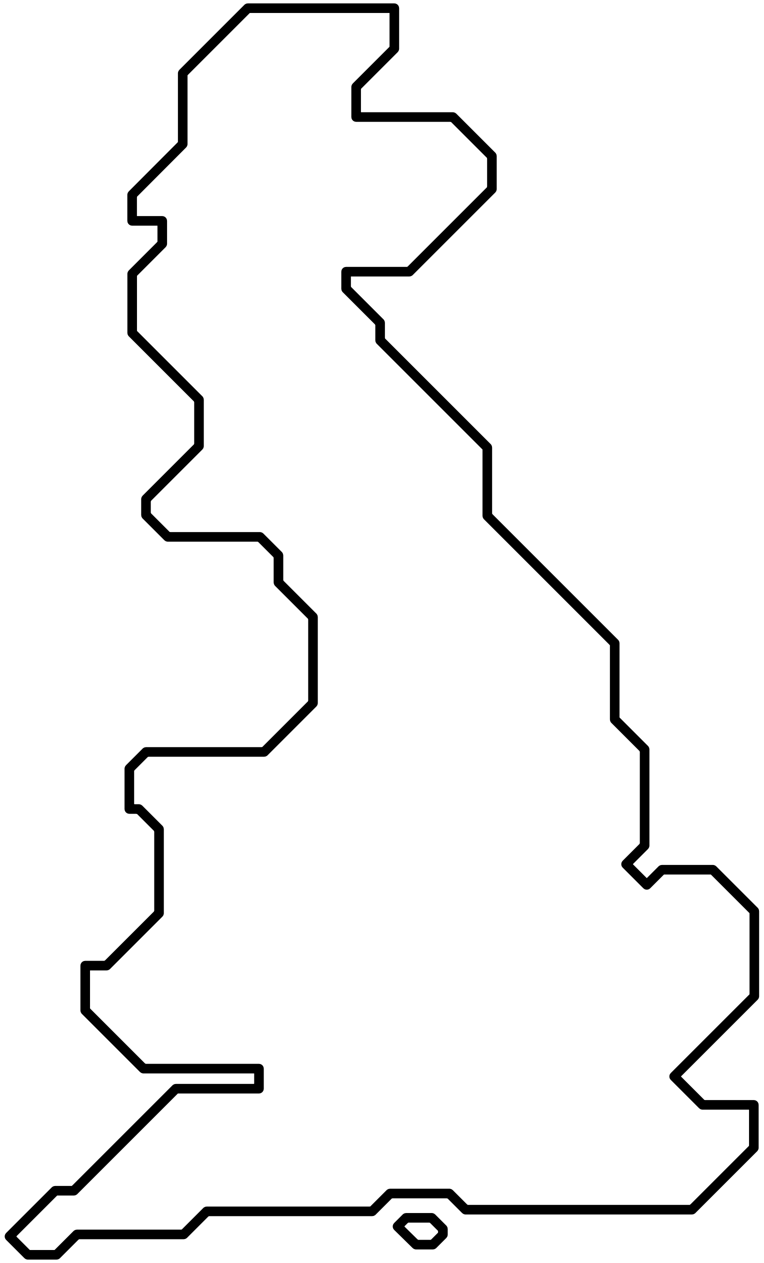
Methods

	Networks	Regions
Lines	[Cabello et al, 2005] [Merrick & Gudmundsson, 2007] [Nöllenburg & Wolff, 2010]	[Buchin et al, 2011] [Cicerone & Cermignani, 2012] **[Buchin et al, to appear]**
Bézier	[Fink et al, 2013]	[Van Goethem et al, 2013]
Circular arcs	[Fink et al, 2014] [Van Goethem et al, 2014]	[Drysdale et al, 2008] [Heimlich & Held, 2008] [Van Goethem et al, 2013] **[Van Goethem et al, 2015]**

Requirements

Few geometric objects

At most k lines (parameter)



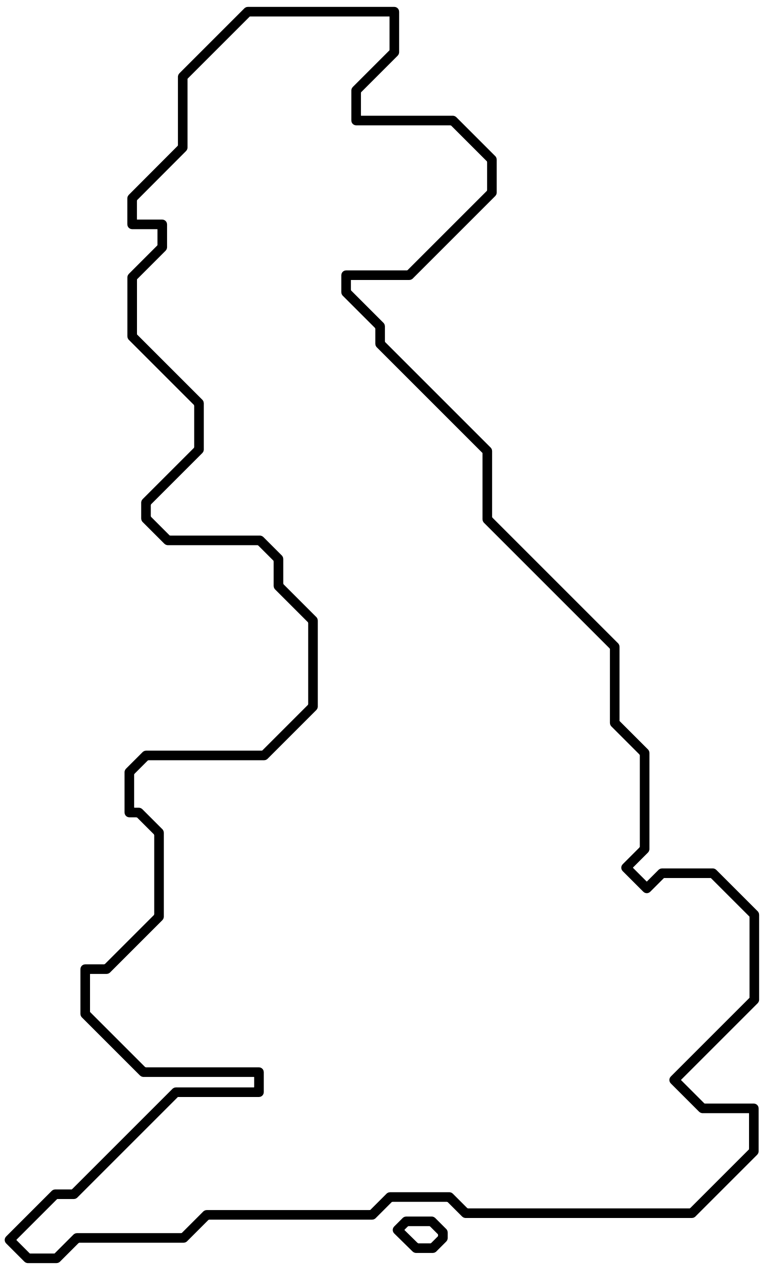
Requirements

Few geometric objects

At most k lines (parameter)

Restricted geometry

Angles in set $\mathcal{C}$ (parameter)



Requirements

Few geometric objects

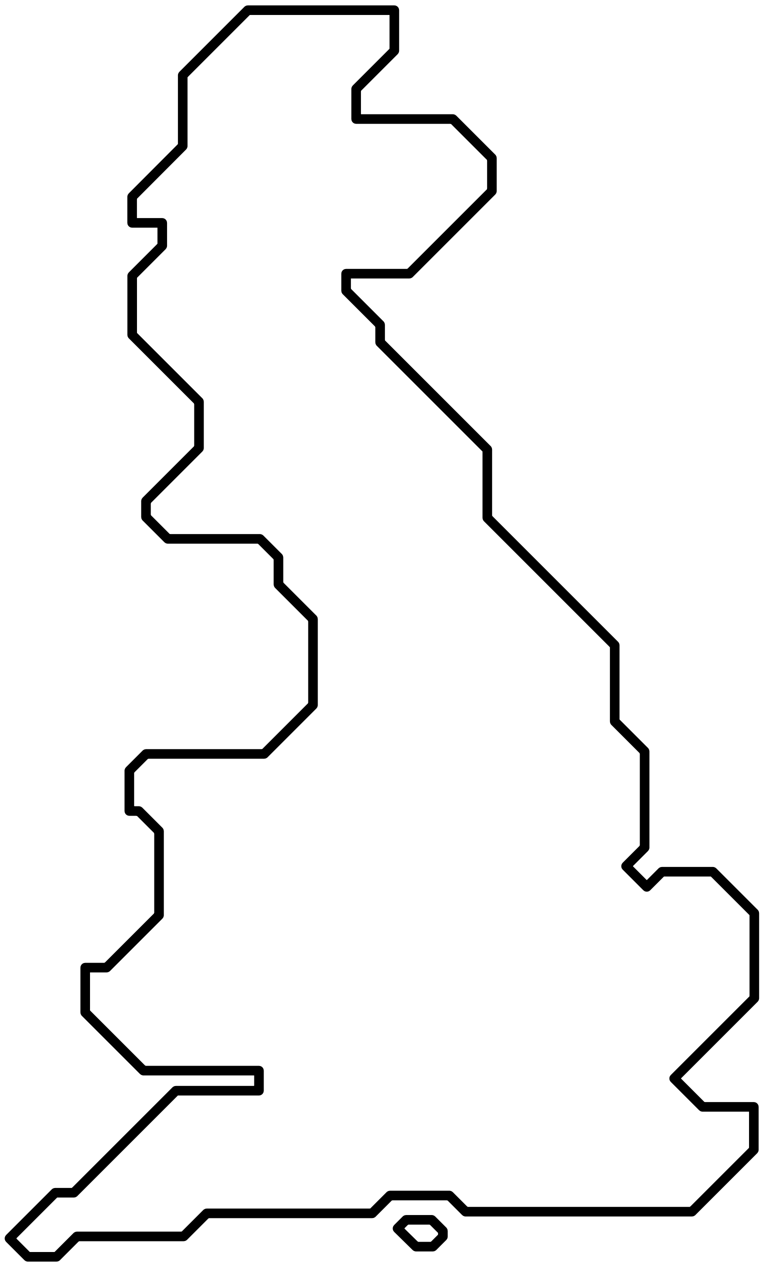
At most k lines (parameter)

Restricted geometry

Angles in set $\mathcal{C}$ (parameter)

Topology

Correct neighbors



Requirements

Few geometric objects

At most k lines (parameter)

Restricted geometry

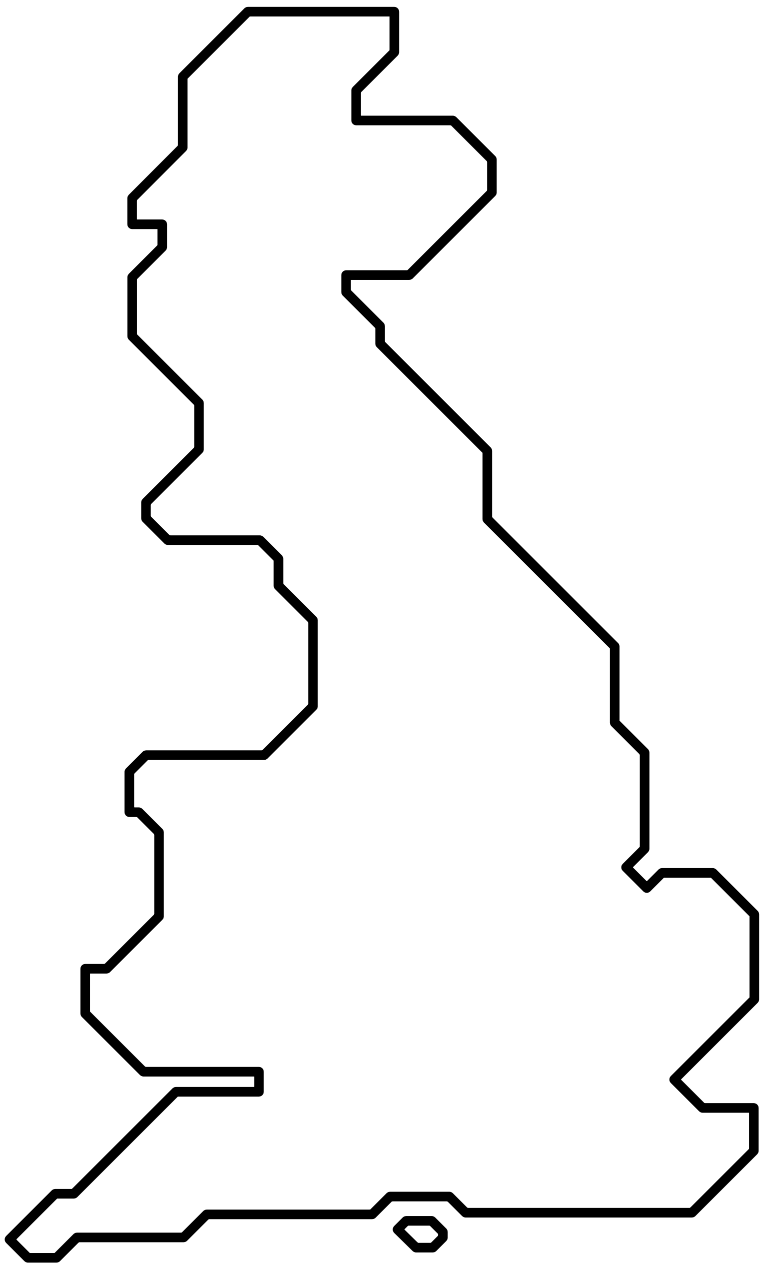
Angles in set $\mathcal{C}$ (parameter)

Topology

Correct neighbors

Resemblance

Area preservation



Requirements

Few geometric objects

At most k lines (parameter)

Restricted geometry

Angles in set $\mathcal{C}$ (parameter)

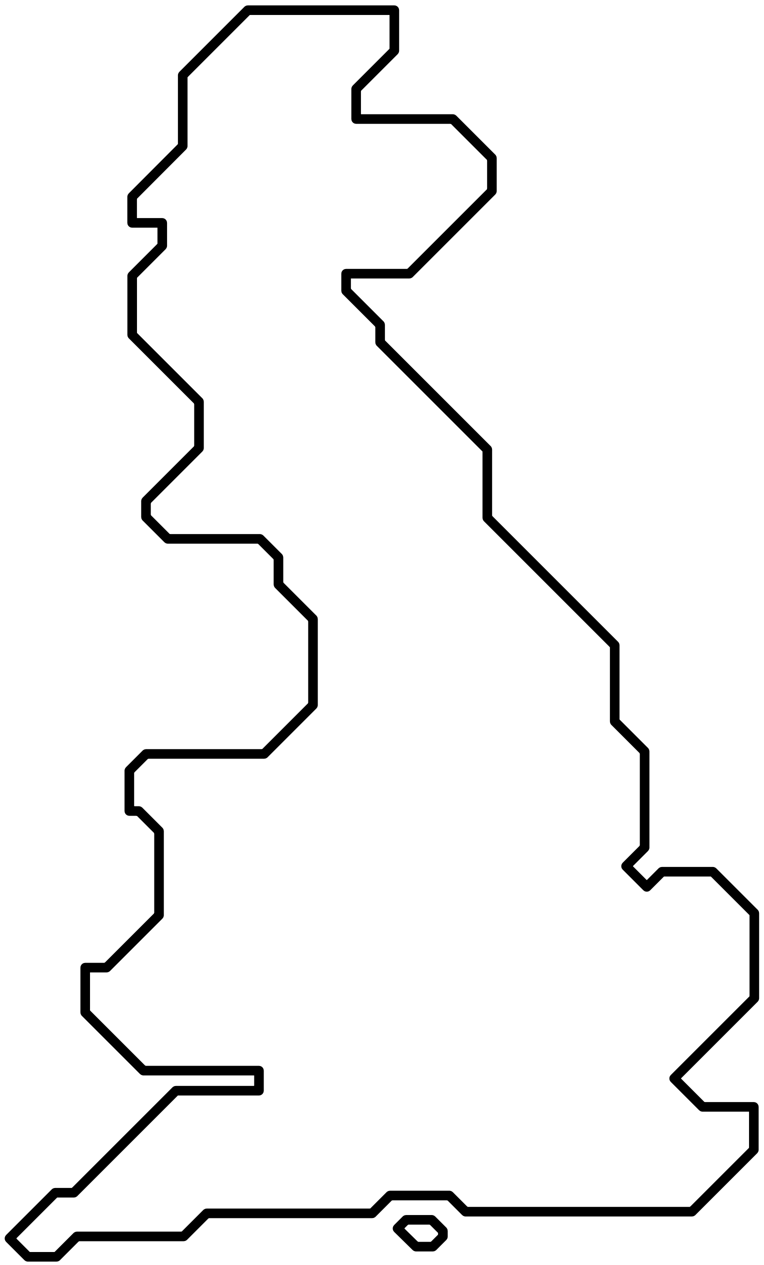
Topology

Correct neighbors

Resemblance

Area preservation

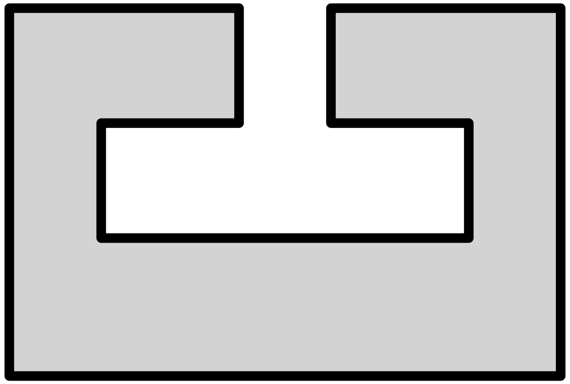
Measure something...?



Formalizing “resemblance”?

Let’s optimize **symmetric difference**

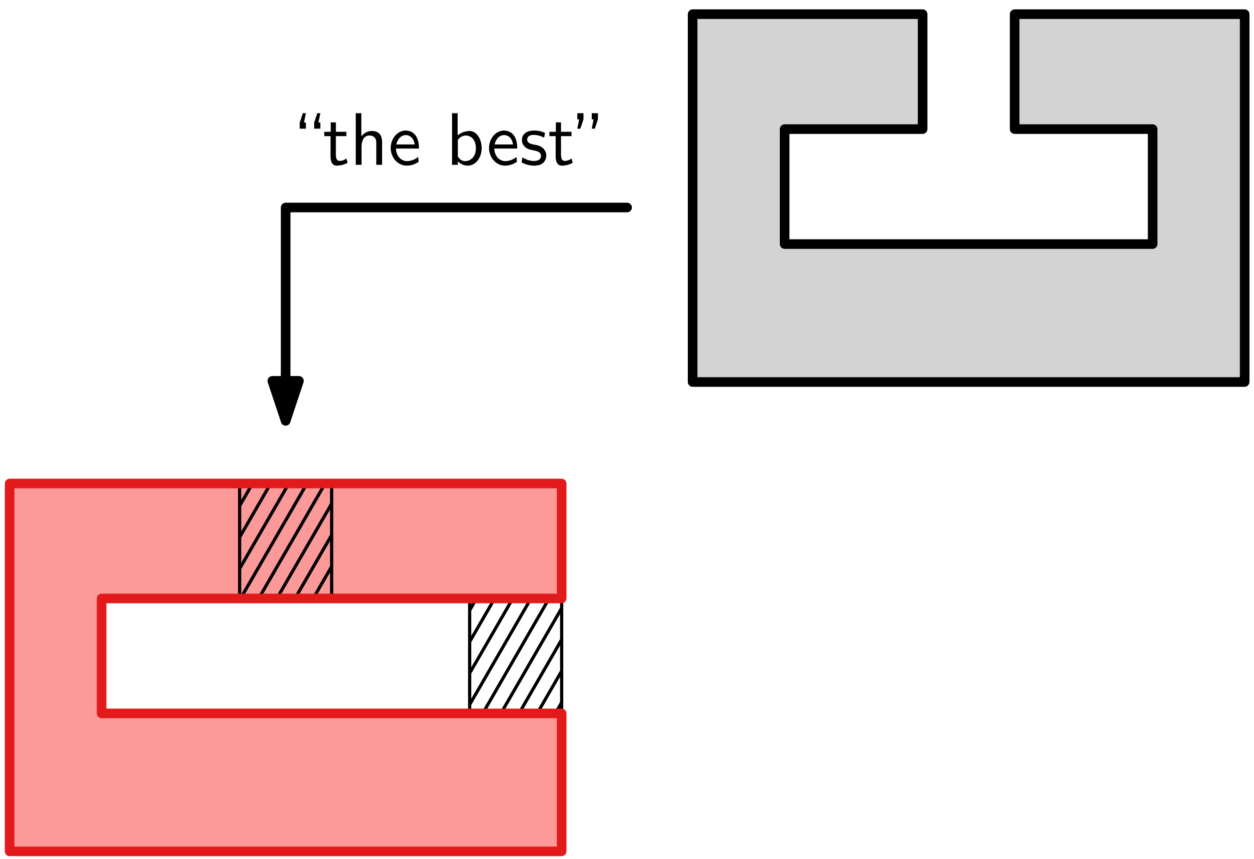
“**area** covered by exactly one polygon”



Formalizing “resemblance”?

Let’s optimize **symmetric difference**

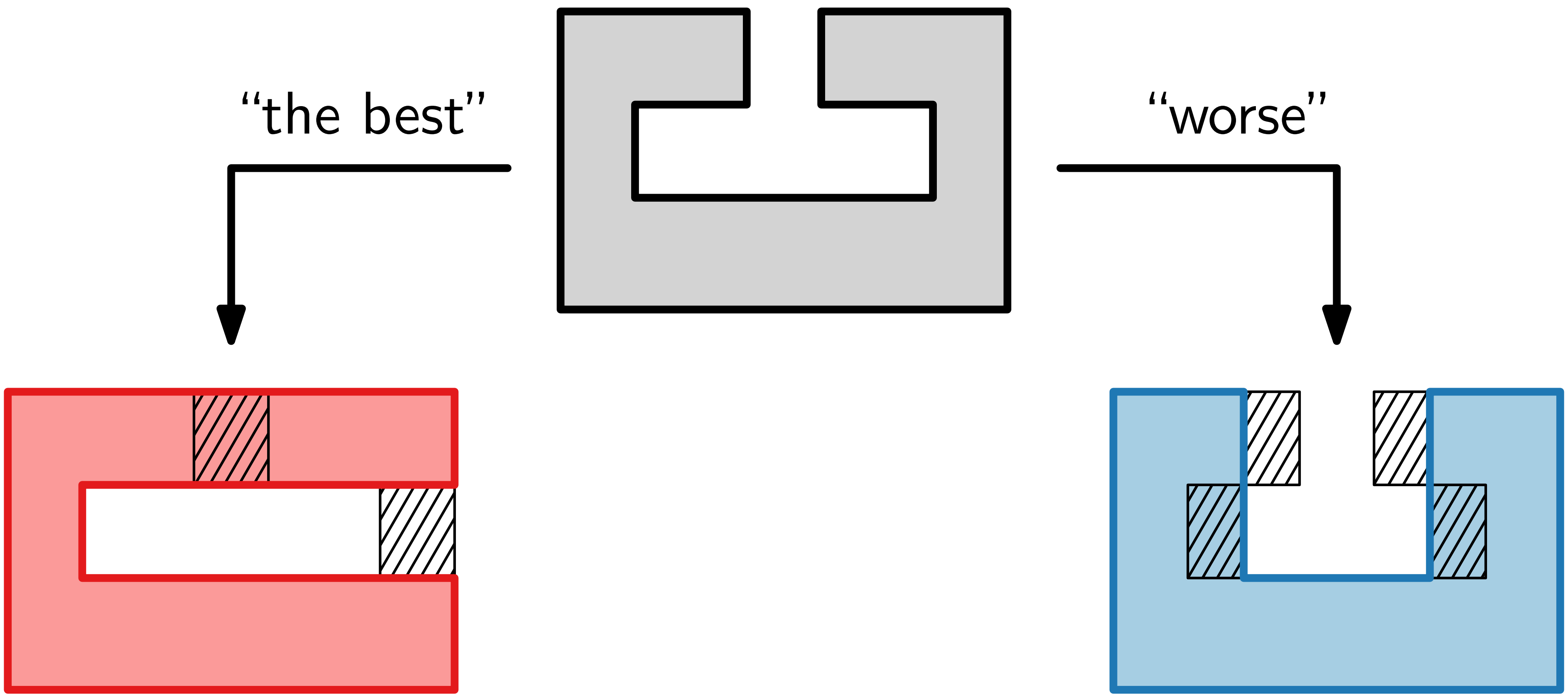
“**area** covered by exactly one polygon”



Formalizing "resemblance"?

Let's optimize **symmetric difference**

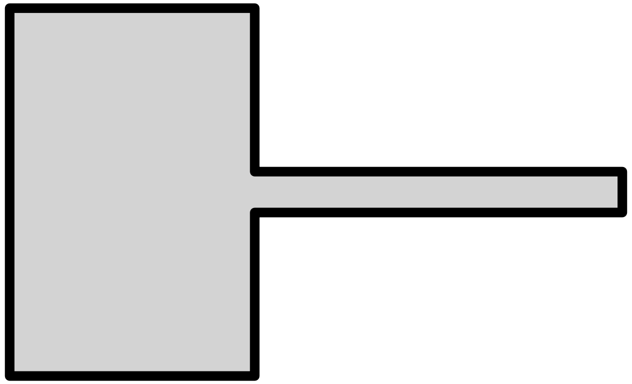
"**area** covered by exactly one polygon"



Formalizing "resemblance"?

Let's try again: **Fréchet distance**

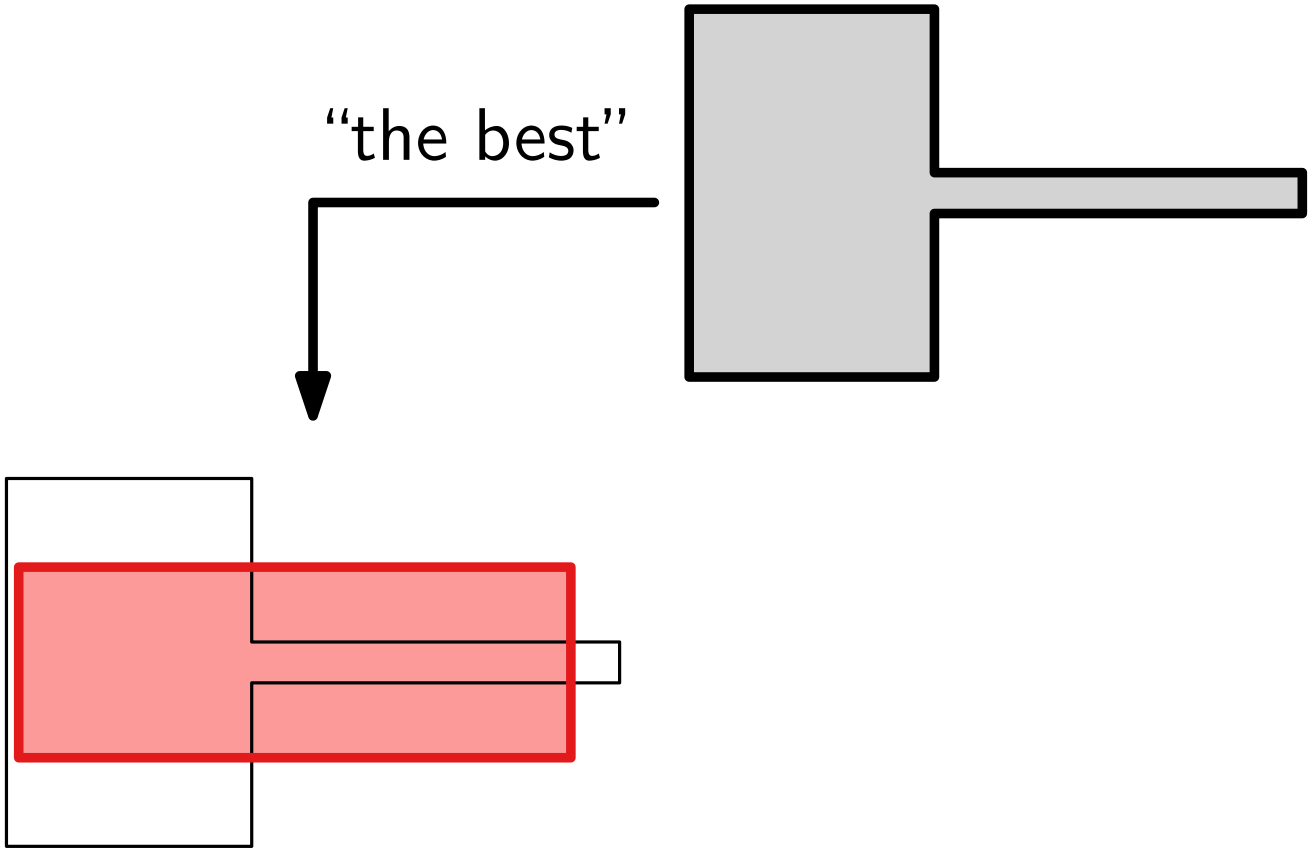
"**longest distance** between boundaries, accounting for continuity"



Formalizing "resemblance"?

Let's try again: **Fréchet distance**

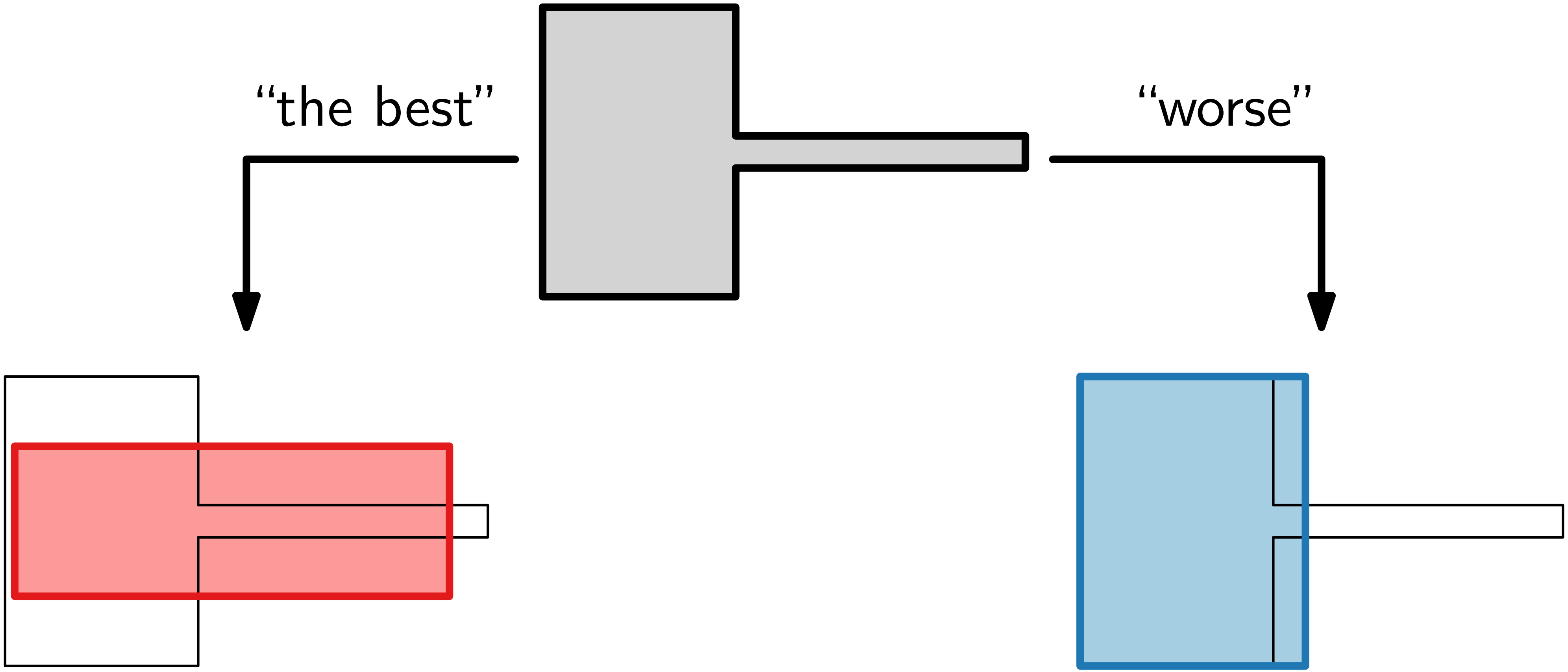
"**longest distance** between boundaries, accounting for continuity"



Formalizing “resemblance”?

Let’s try again: **Fréchet distance**

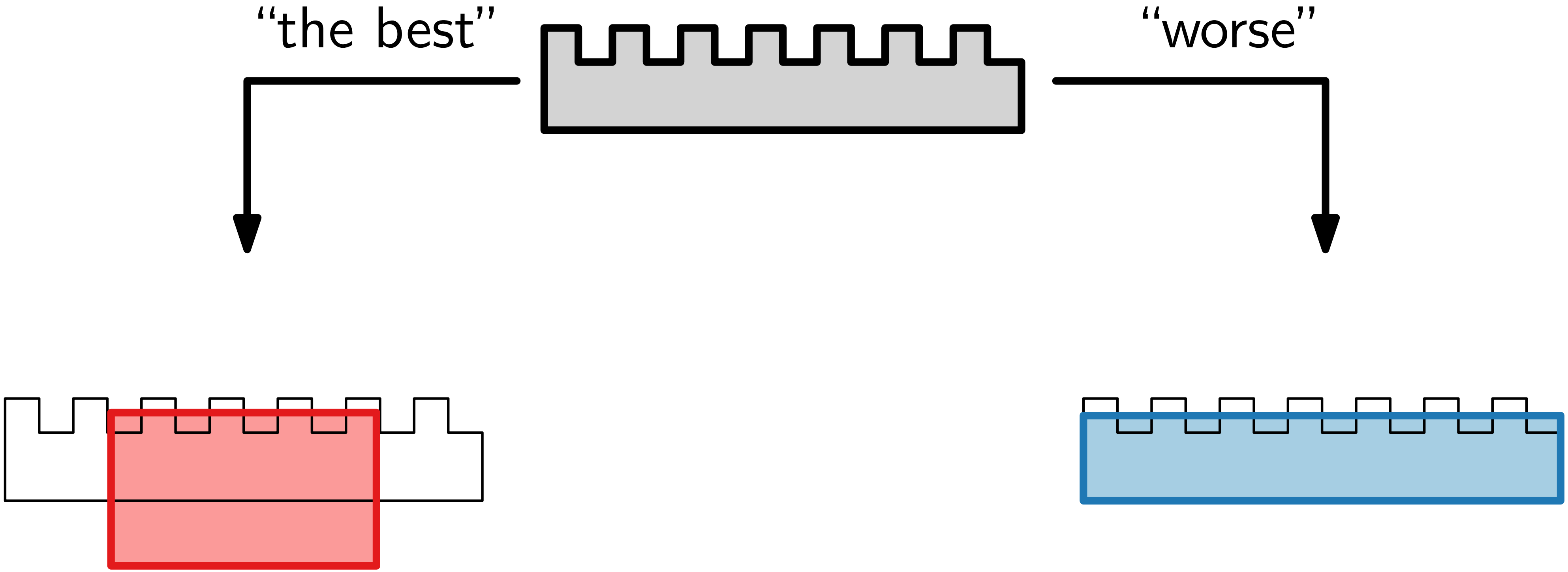
“**longest distance** between boundaries, accounting for continuity”



Formalizing "resemblance"?

Once more: **cyclic dynamic time warp distance**

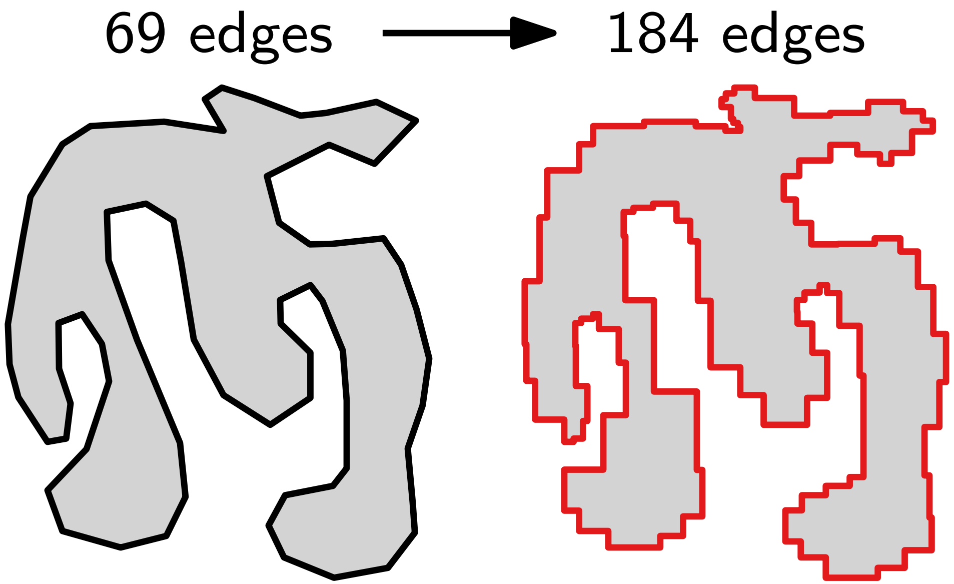
"**Sum of distances** between vertices, accounting for continuity"



Algorithm

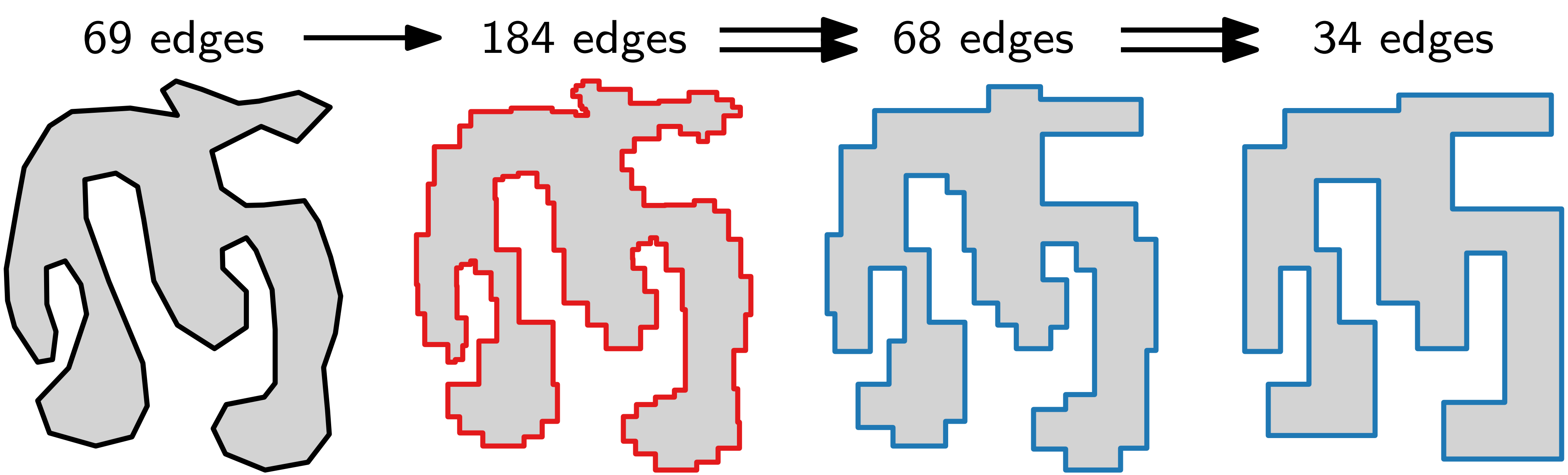
Algorithm

1. `restrict angles to` $\mathcal{C}$



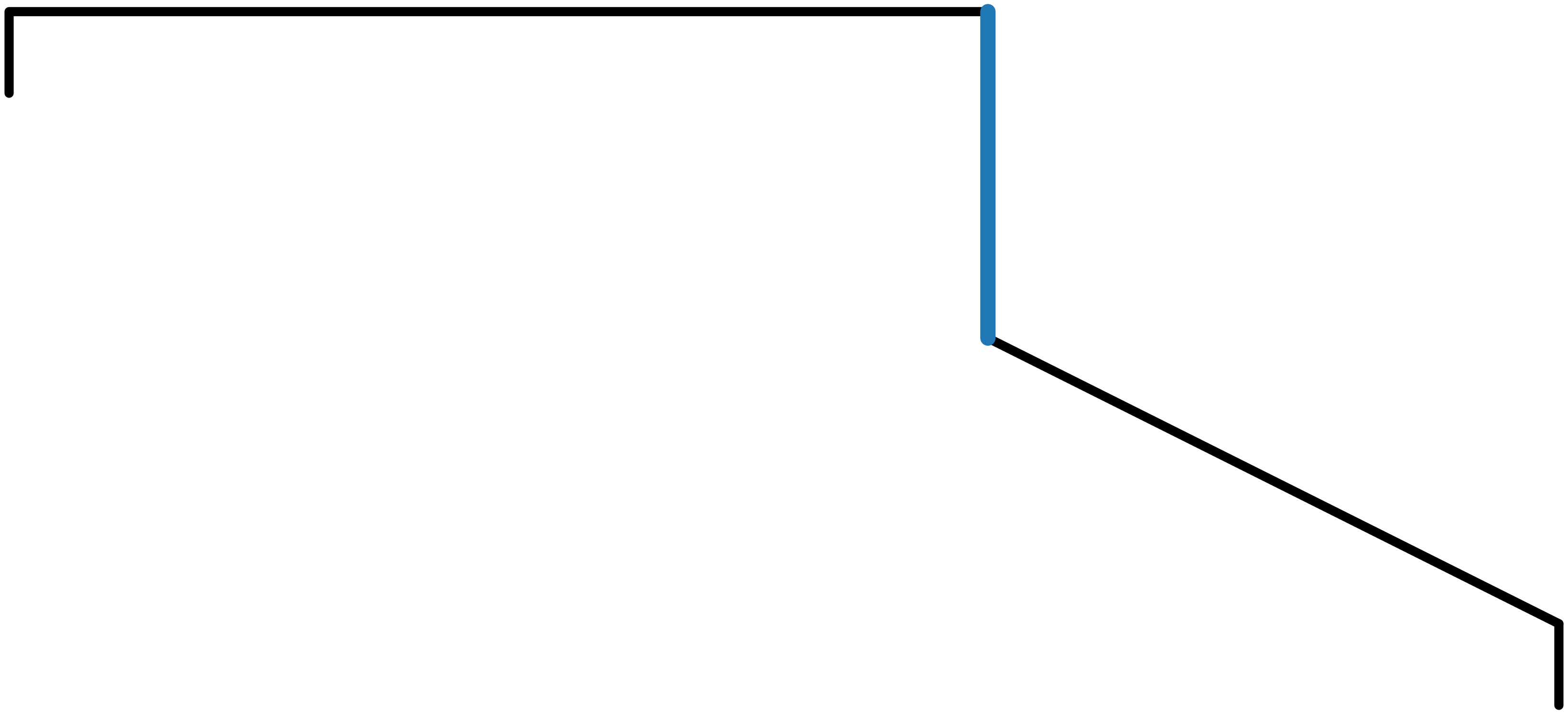
Algorithm

1. restrict angles to $\mathcal{C}$
2. repeat
3.   perform a pair of edge-moves
4. until at most k lines



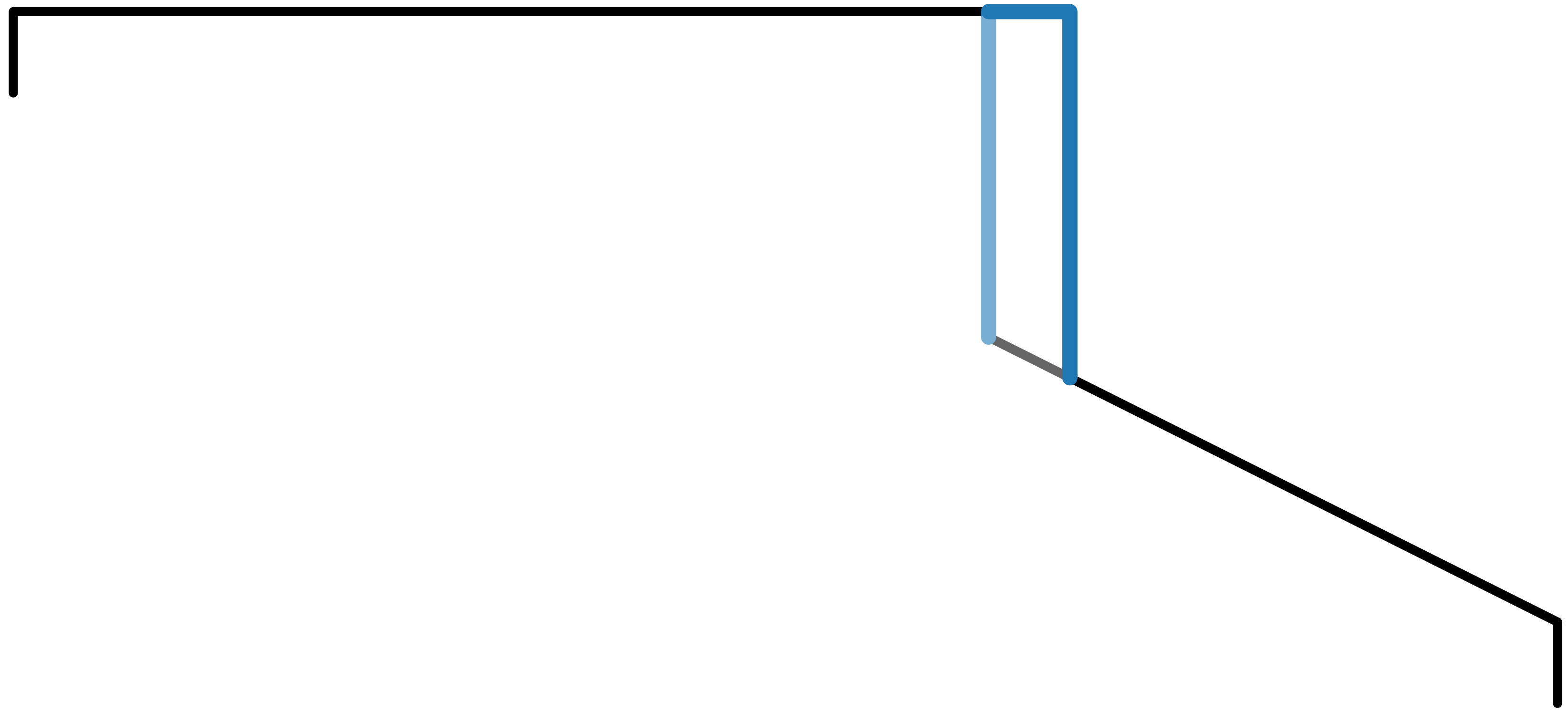
Edge-moves

3. perform a pair of edge-moves



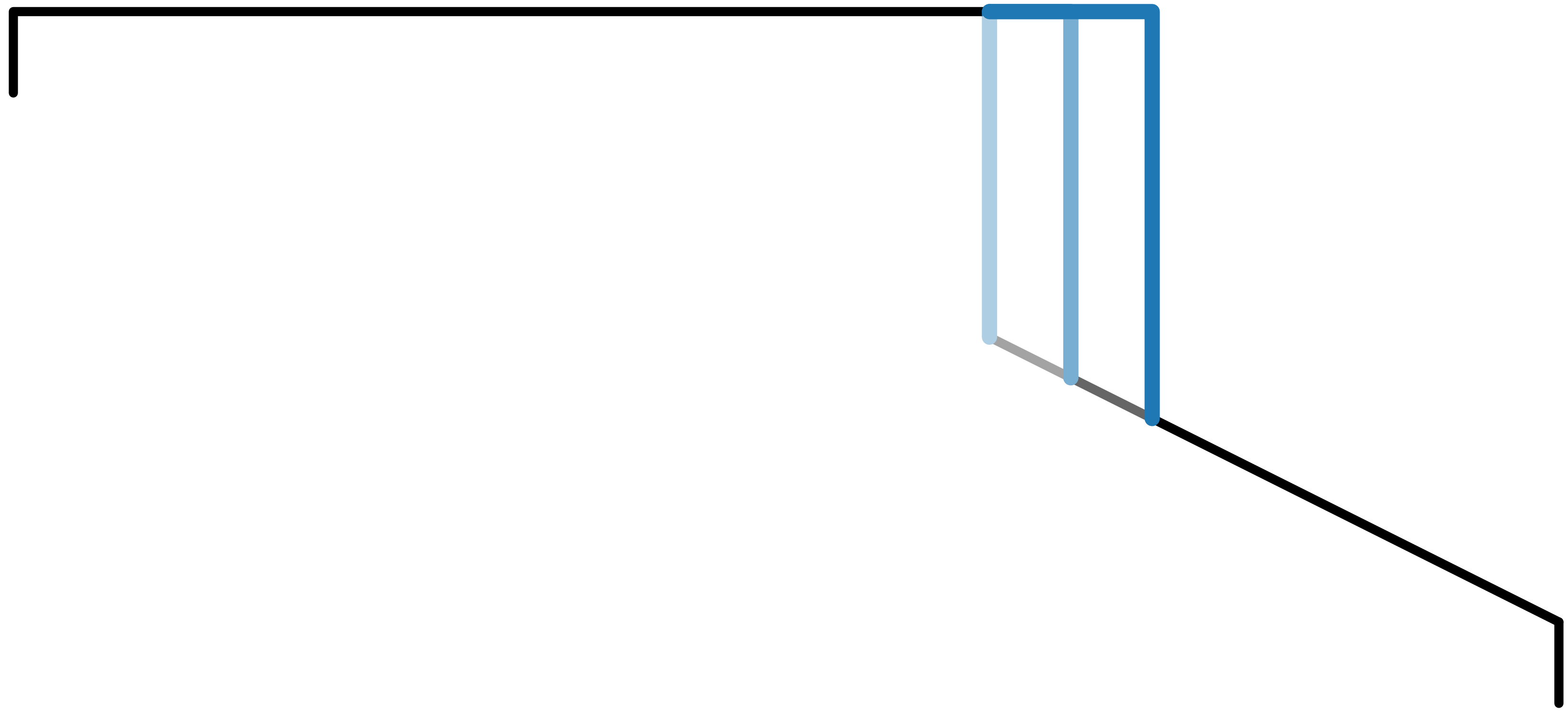
Edge-moves

3. perform a pair of edge-moves



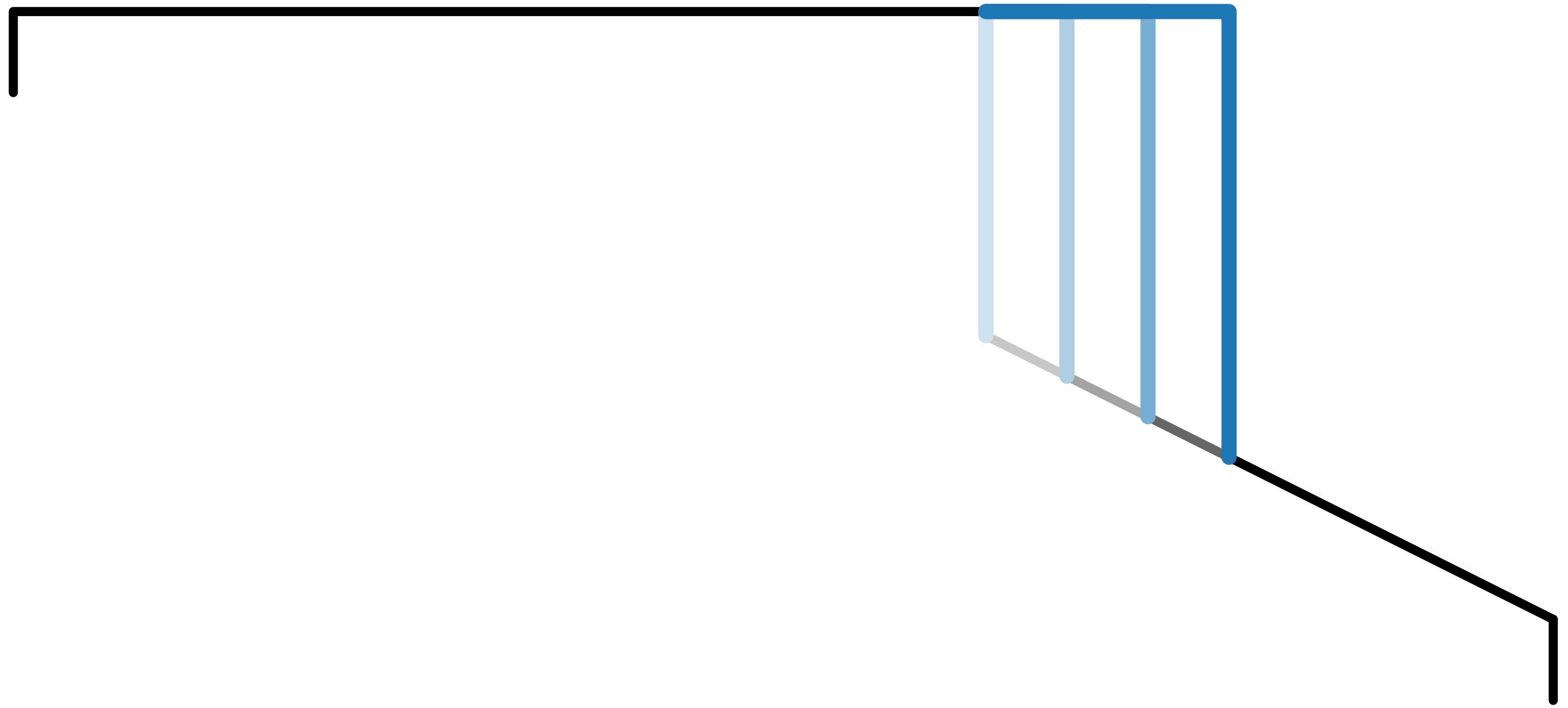
Edge-moves

3. perform a pair of edge-moves



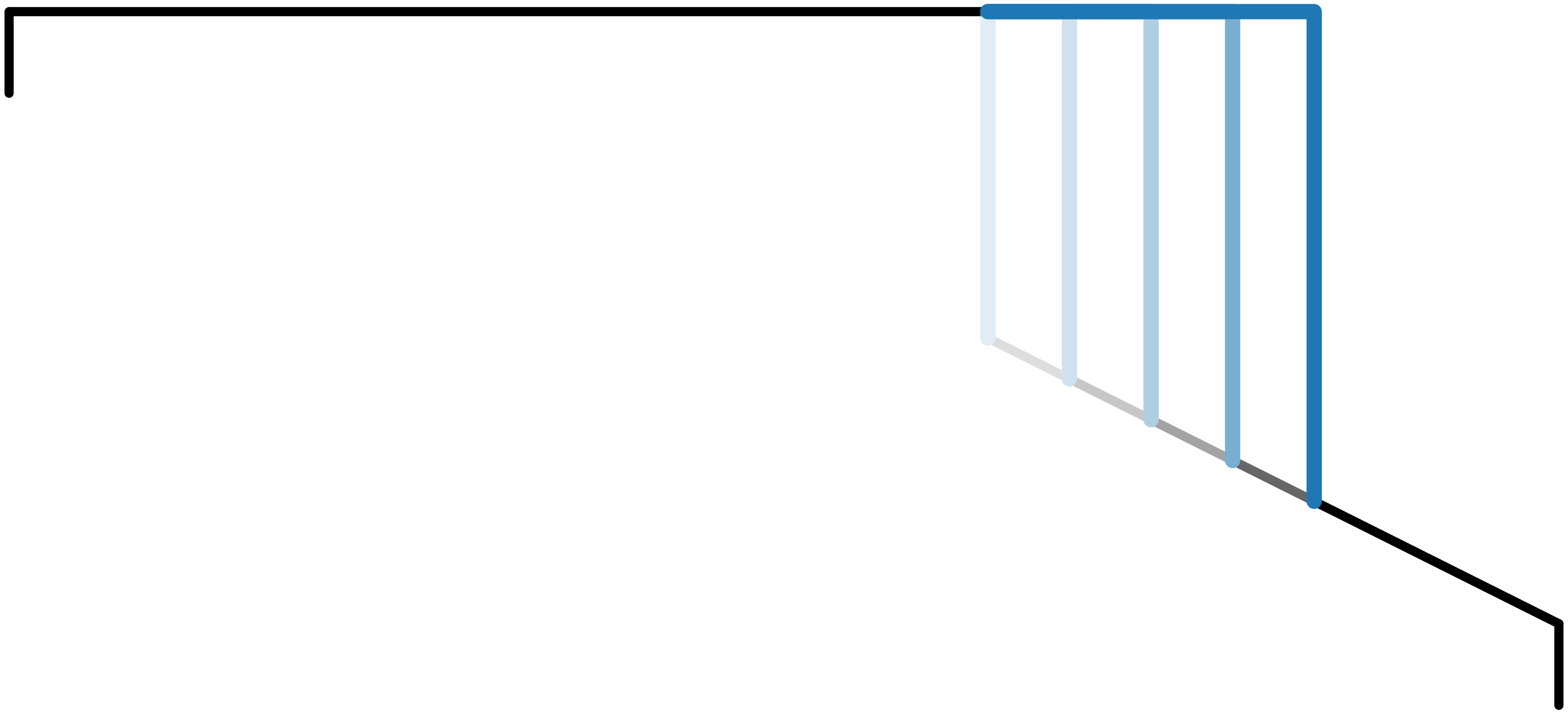
Edge-moves

3. perform a pair of edge-moves



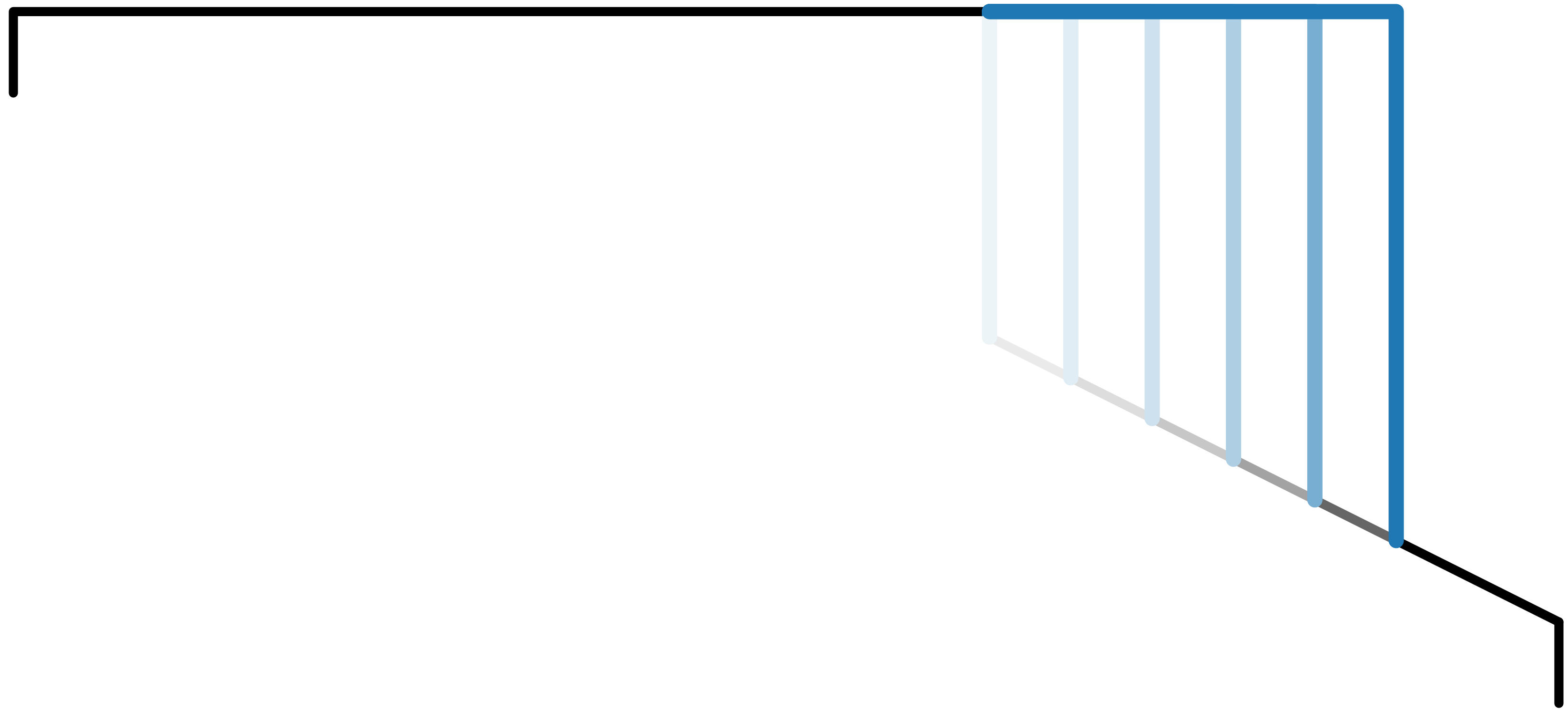
Edge-moves

3. perform a pair of edge-moves



Edge-moves

3. perform a pair of edge-moves



Edge-moves

3. perform a pair of edge-moves

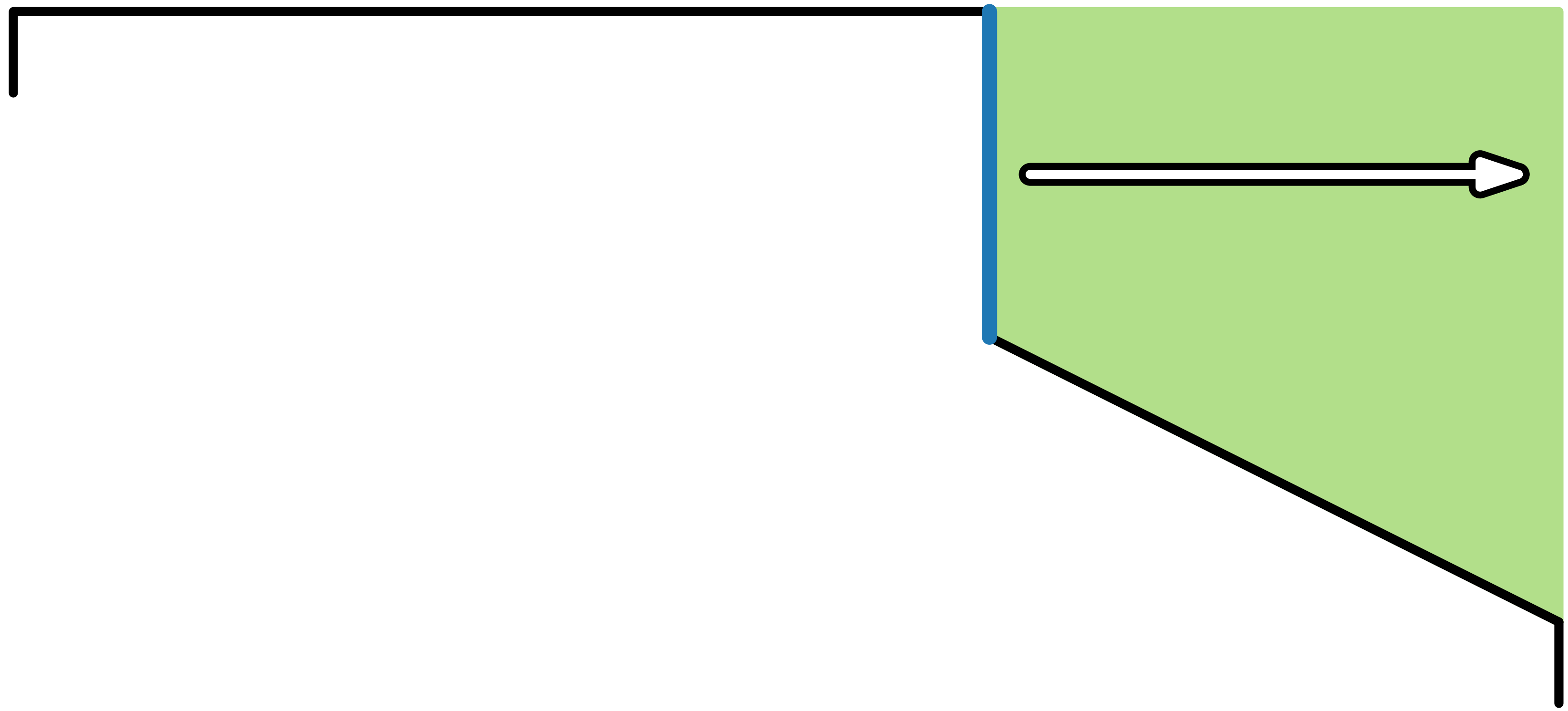
Edge-moves

3. perform a pair of edge-moves

Edge-moves

3. perform a pair of edge-moves

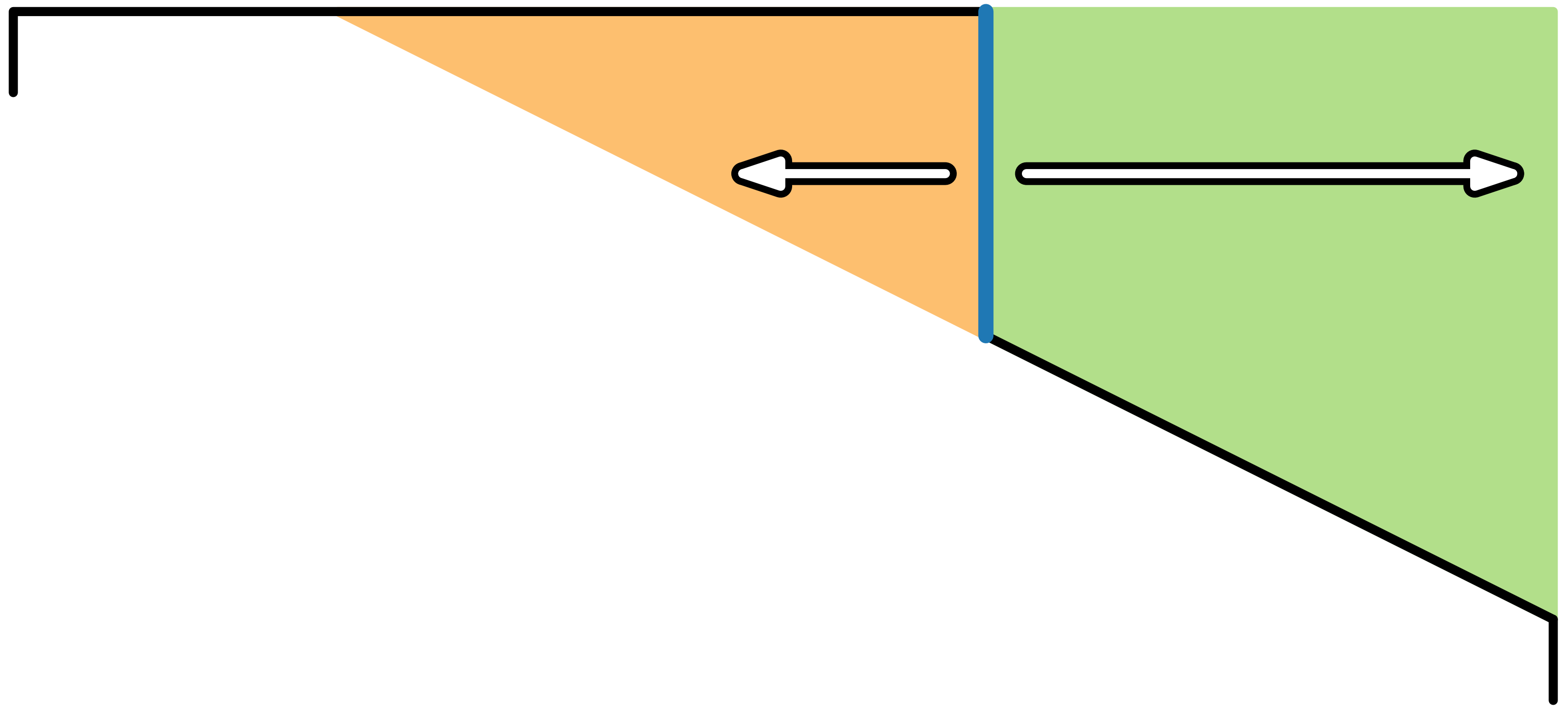
Preserves **topology** and **angles**



Edge-moves

3. perform a pair of edge-moves

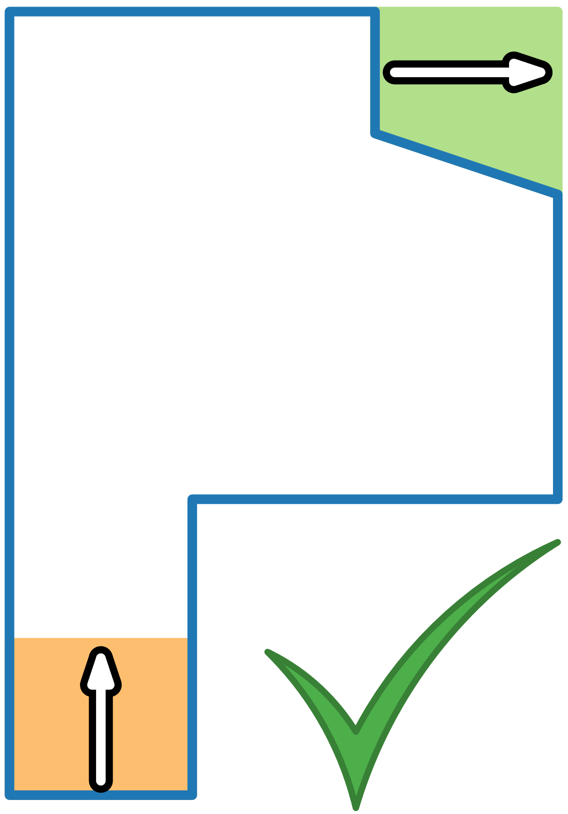
Preserves **topology** and **angles**



Edge-moves

3. `perform a pair of edge-moves`

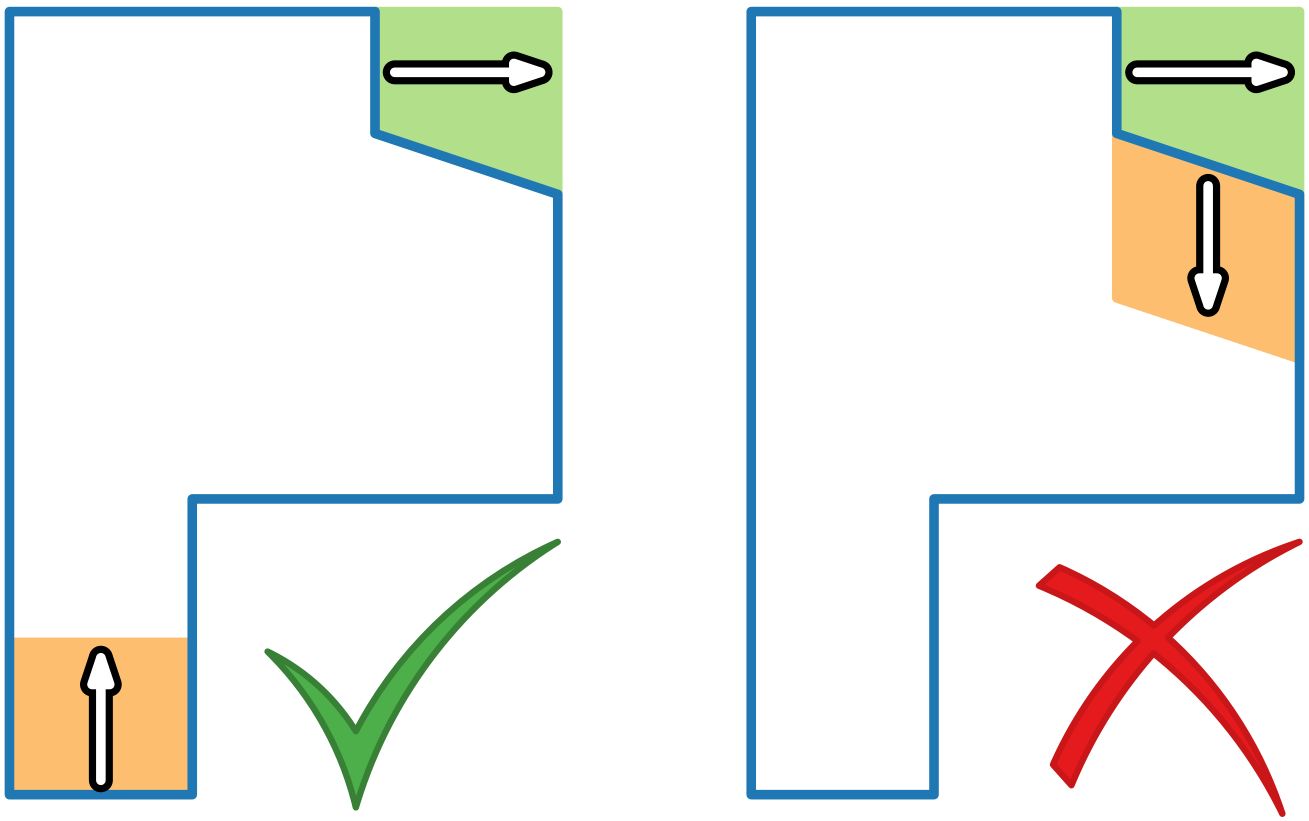
Use pairs to preserve **area**, but avoid conflicts



Edge-moves

3. perform a pair of edge-moves

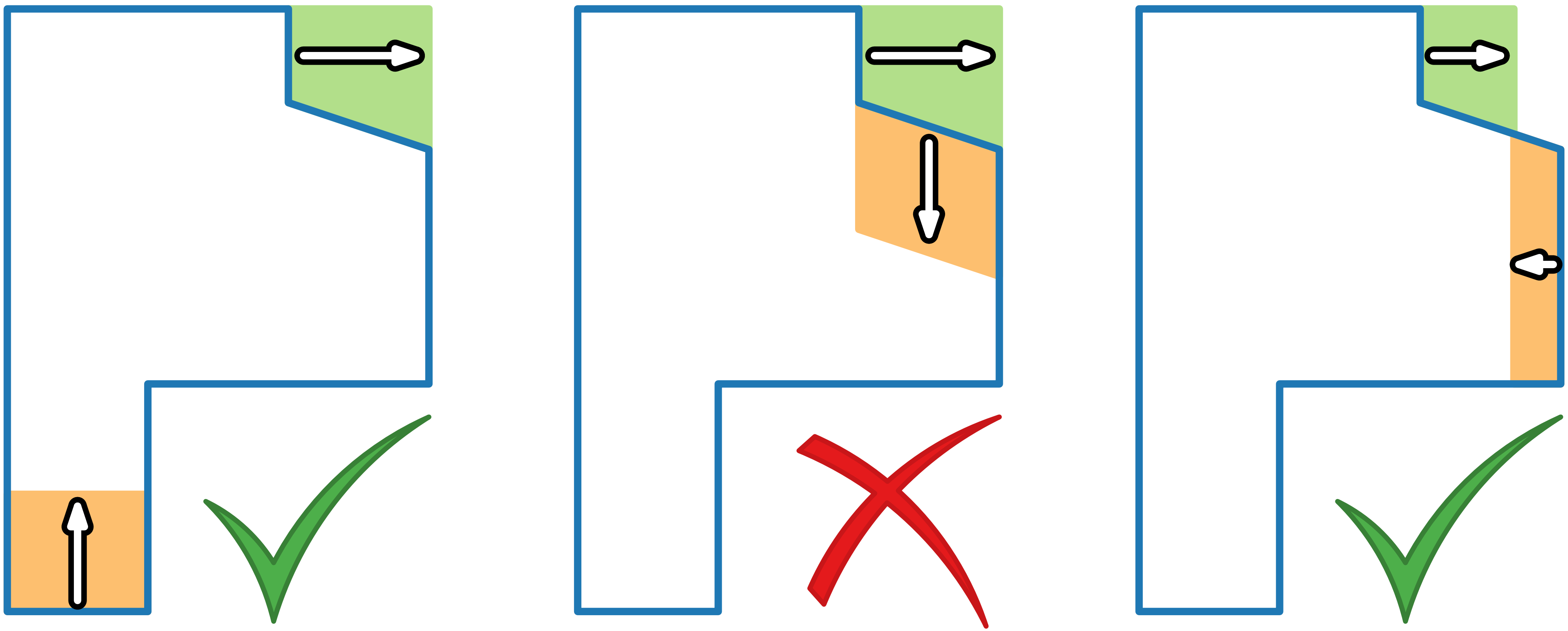
Use pairs to preserve **area**, but avoid conflicts



Edge-moves

3. perform a pair of edge-moves

Use pairs to preserve **area**, but avoid conflicts



Edge-moves

```
3.    perform a pair of edge-moves
```

But **which pair** do we pick?

Smallest area (symmetric difference)

Compensate with nearest along boundary

Termination

```
4. until at most k lines
```

Can we always reach k?

Termination

`4.` `until` `at most` k `lines`

Can we always reach k?

Theorem.
Any nonconvex polygon admits a pair of edge-moves.

$\Rightarrow$ For polygons, we can always reach $2|\mathcal{C}|$

Termination

```
4. until at most k lines
```

Can we always reach k?

Theorem.
Any nonconvex polygon admits a pair of edge-moves.

$\Rightarrow$ For polygons, we can always reach $2|\mathcal{C}|$

How fast is the algorithm?

Naive: $O(n^3)$

Termination

`4. until at most` k `lines`

Can we always reach k?

Theorem.
Any nonconvex polygon admits a pair of edge-moves.

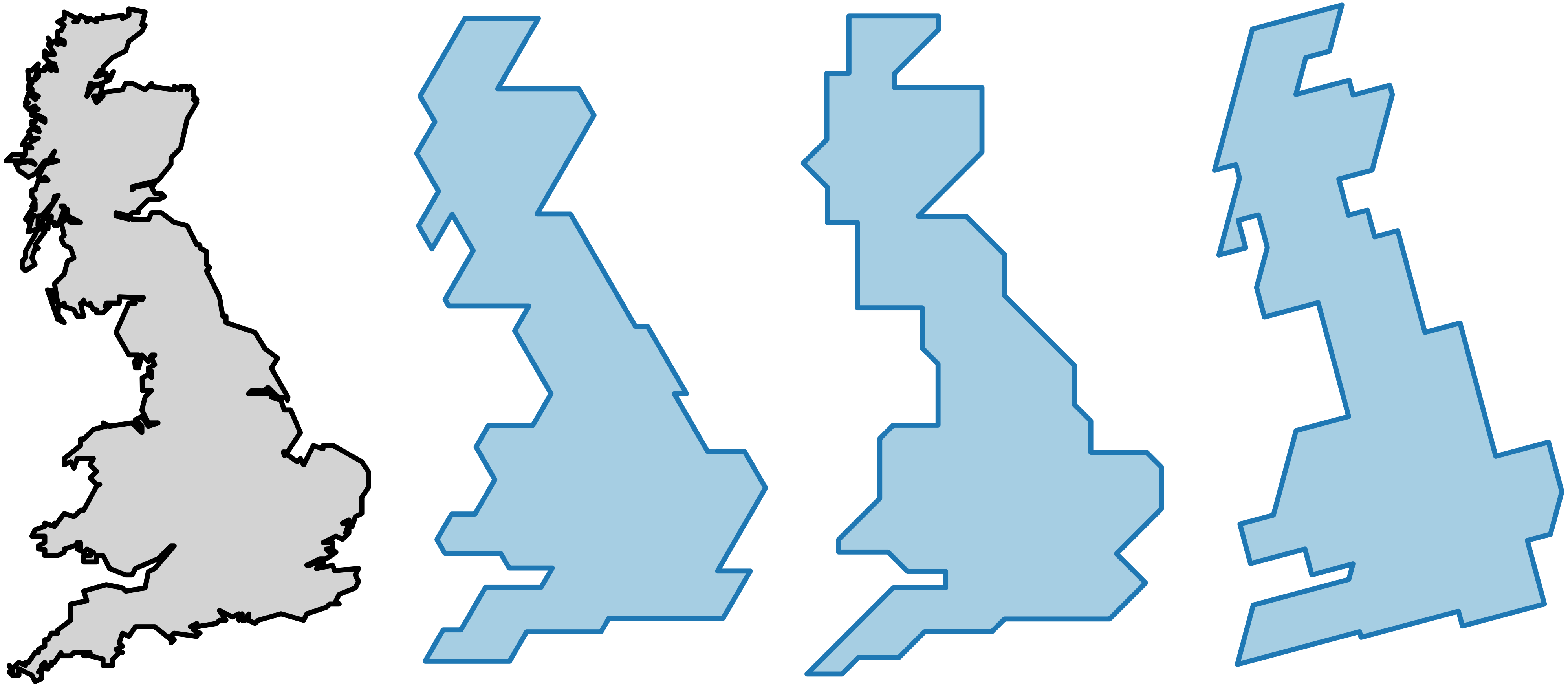
$\Rightarrow$ For polygons, we can always reach $2|\mathcal{C}|$

How fast is the algorithm?

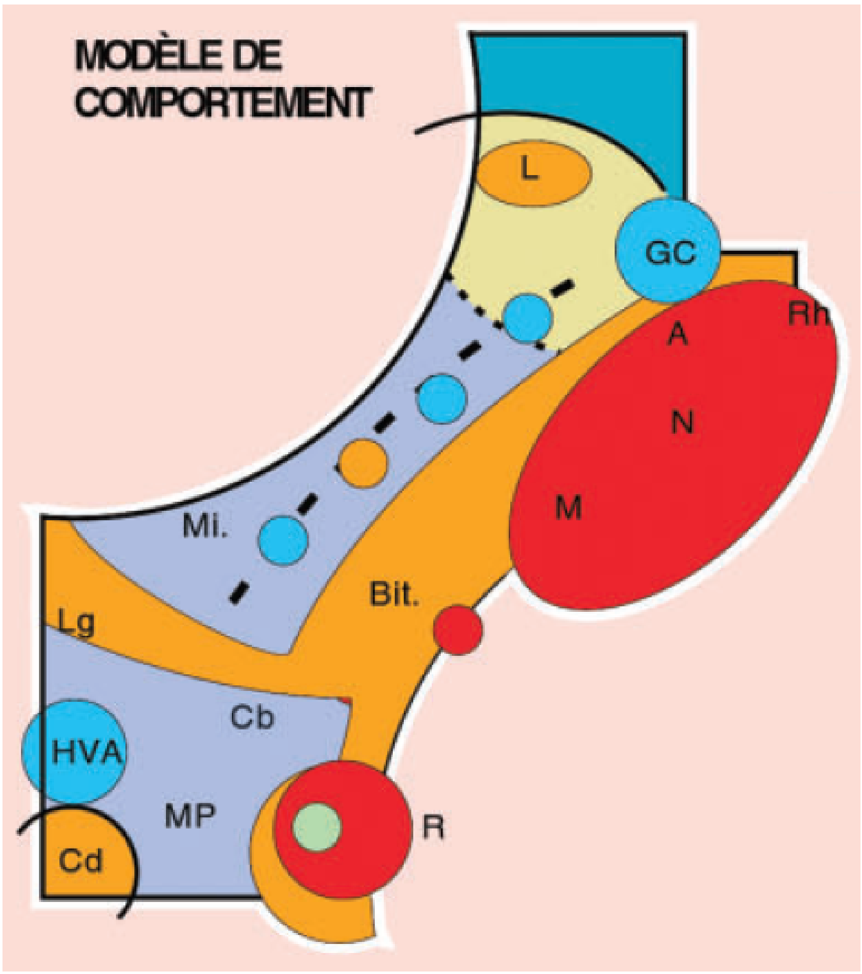
Naive: $O(n^3)$

Using locality of change: $O(n^2)$

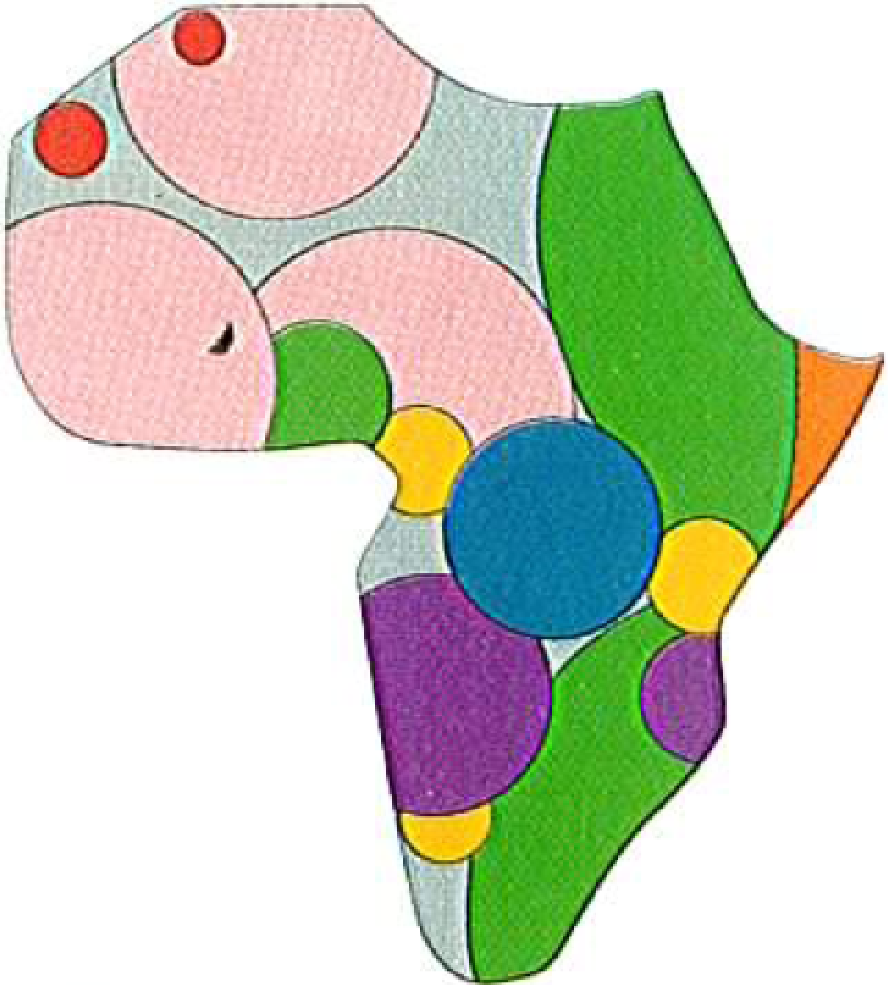
Schematization styles



Do we really need lines?



[Brunet, 1991]



[Brunet & Dollfus, 1991]

Circular arcs

Change edge-moves to **replacements**

- Replace sequence of arcs by fewer arcs
- Turn lines into arcs

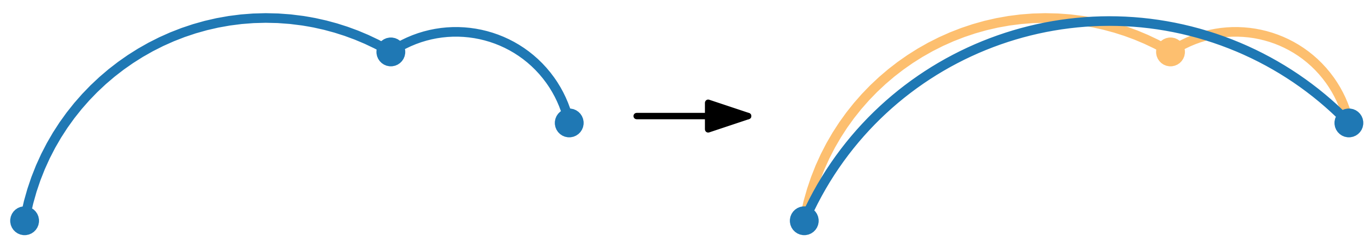
Circular arcs

Change edge-moves to **replacements**

Replace sequence of arcs by fewer arcs

Turn lines into arcs

2-to-1

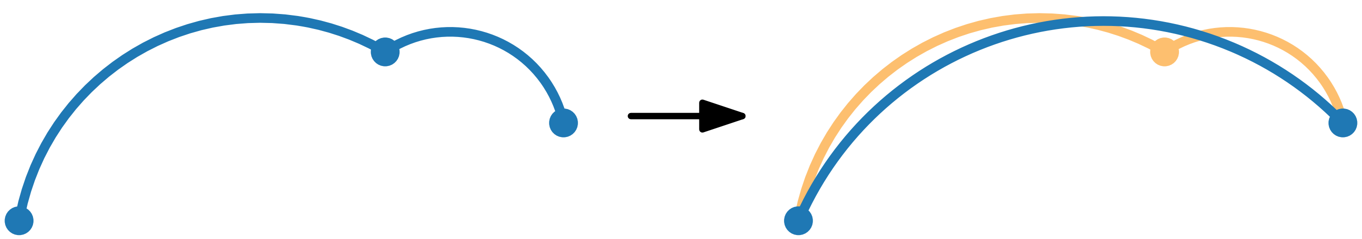


Circular arcs

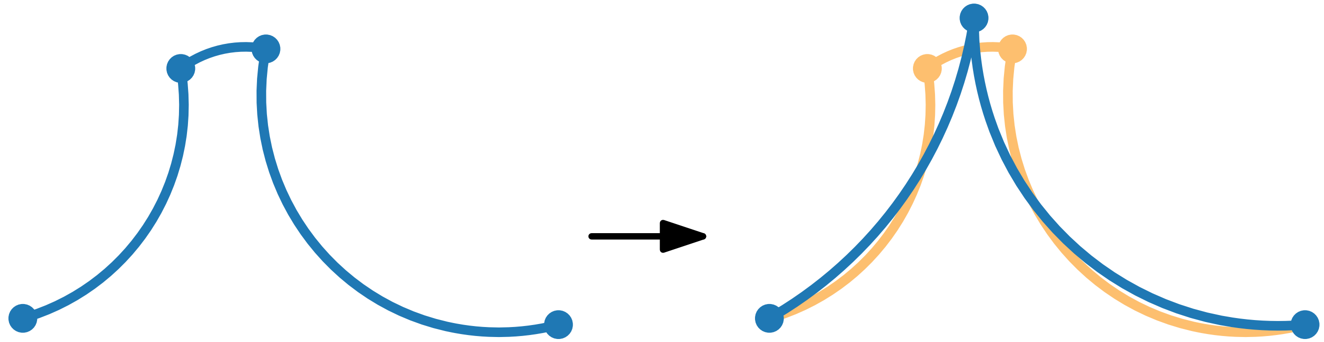
Change edge-moves to **replacements**

- Replace sequence of arcs by fewer arcs
- Turn lines into arcs

2-to-1

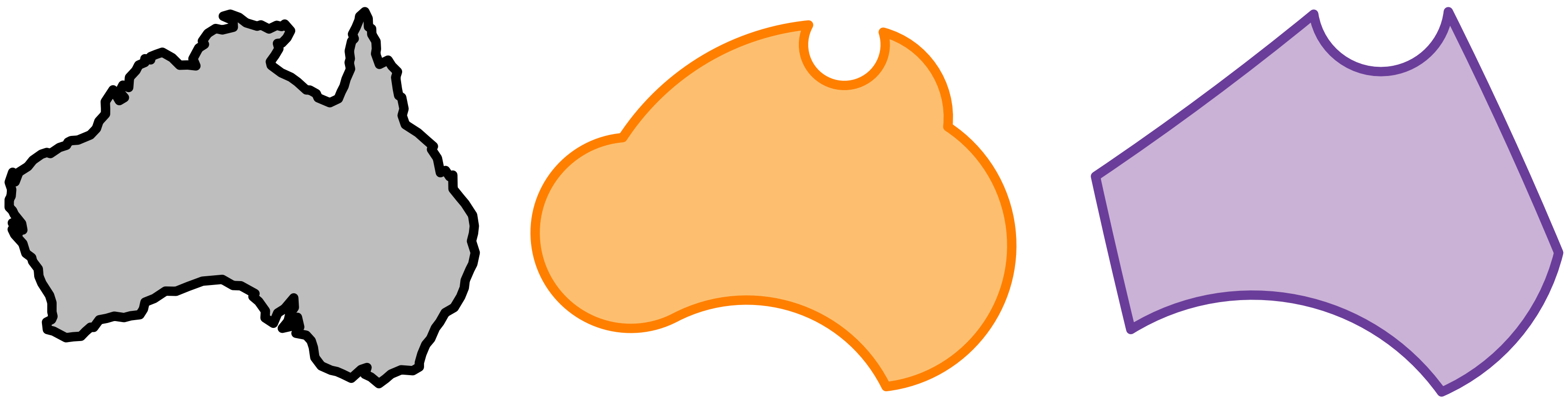


3-to-2



Curviness

Control **curviness**

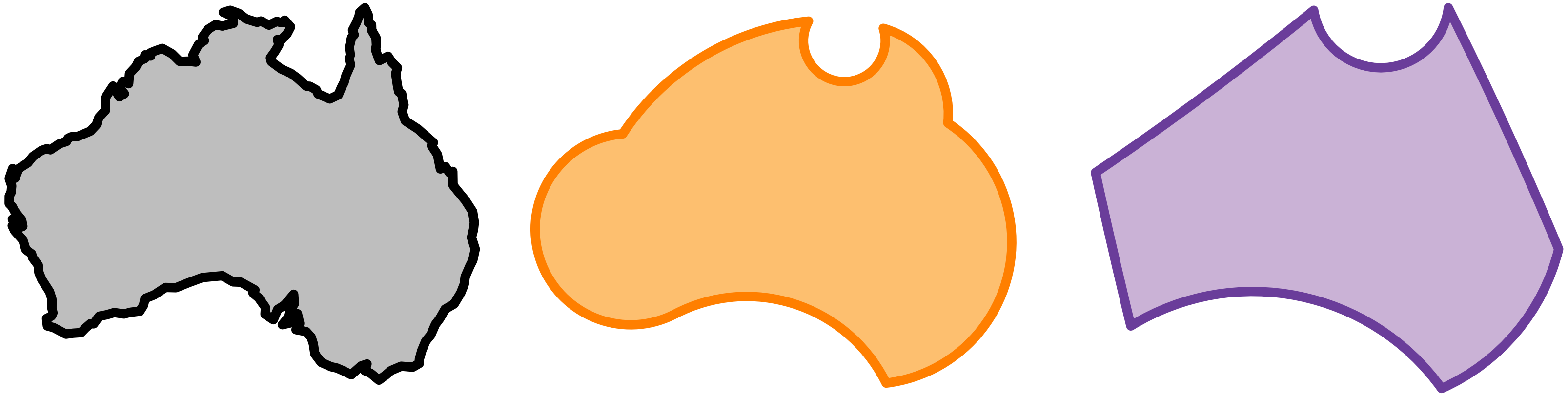


Curviness

Control **curviness**

Central angle α as weight

Gives **curved**, **regular** and **flat** style

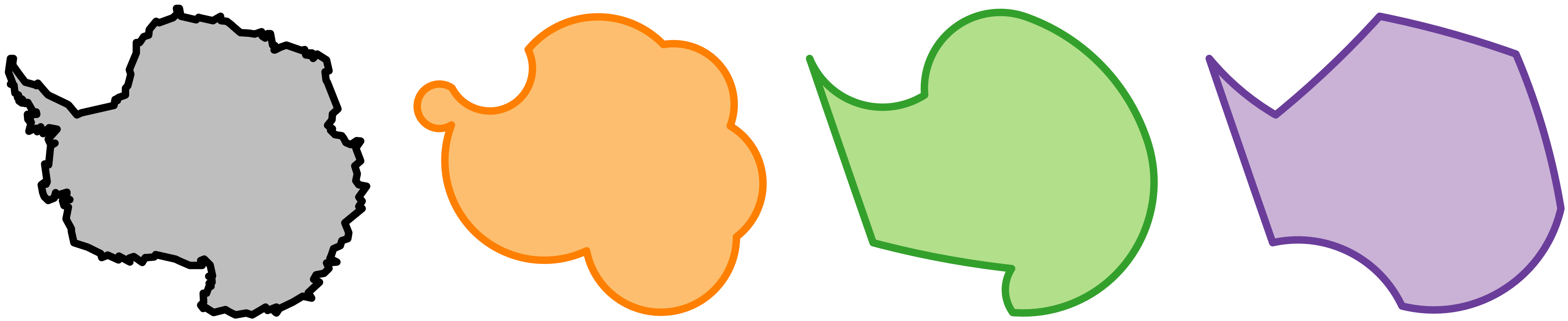


Curviness

Control **curviness**

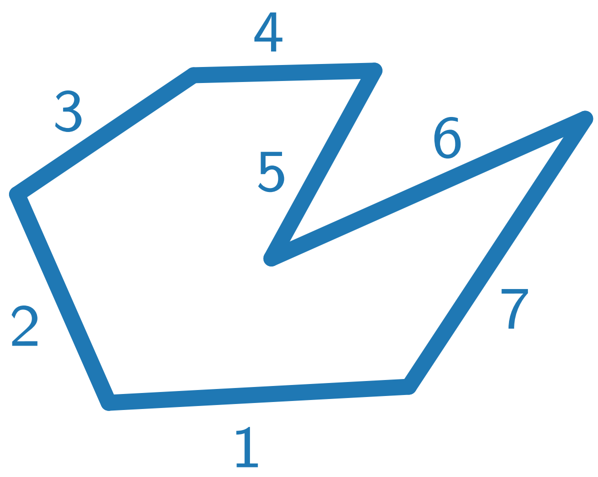
Central angle α as weight

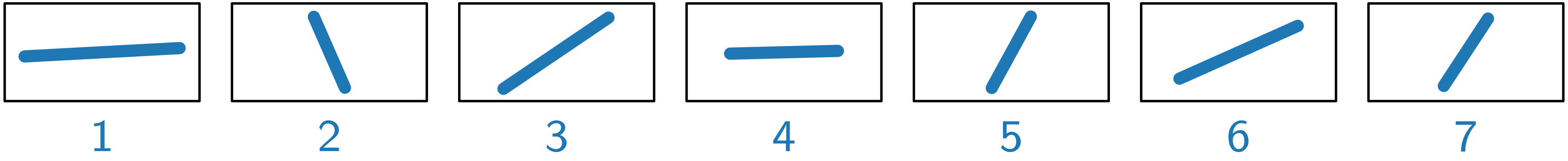
Gives **curved**, **regular** and **flat** style



Recovering solutions

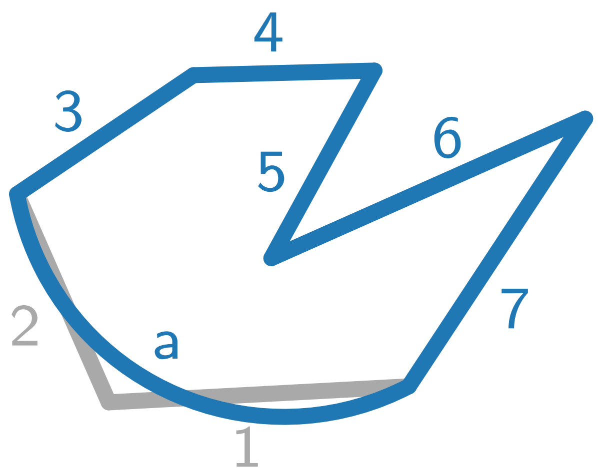
Run once, obtain all solutions

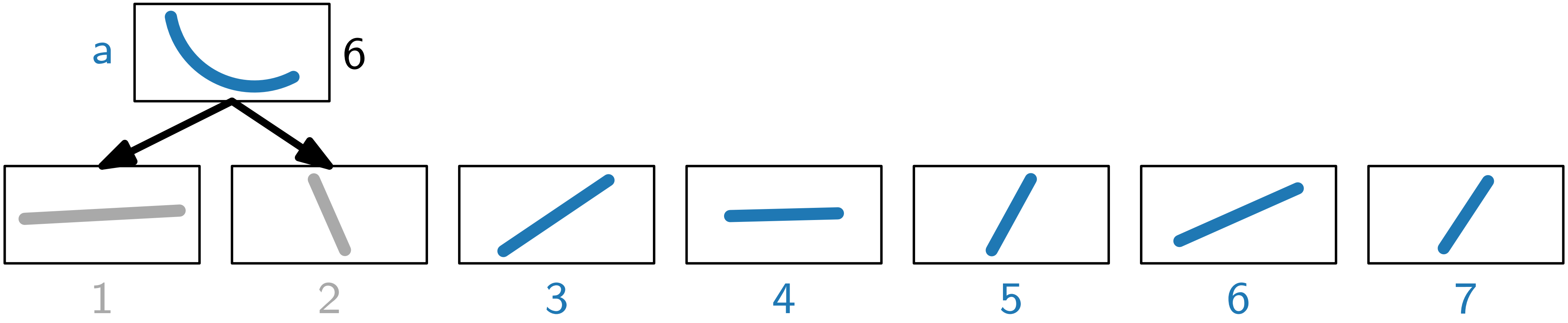




Recovering solutions

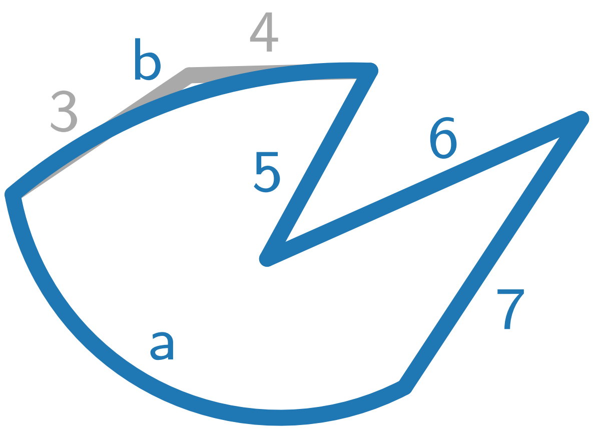
Run once, obtain all solutions

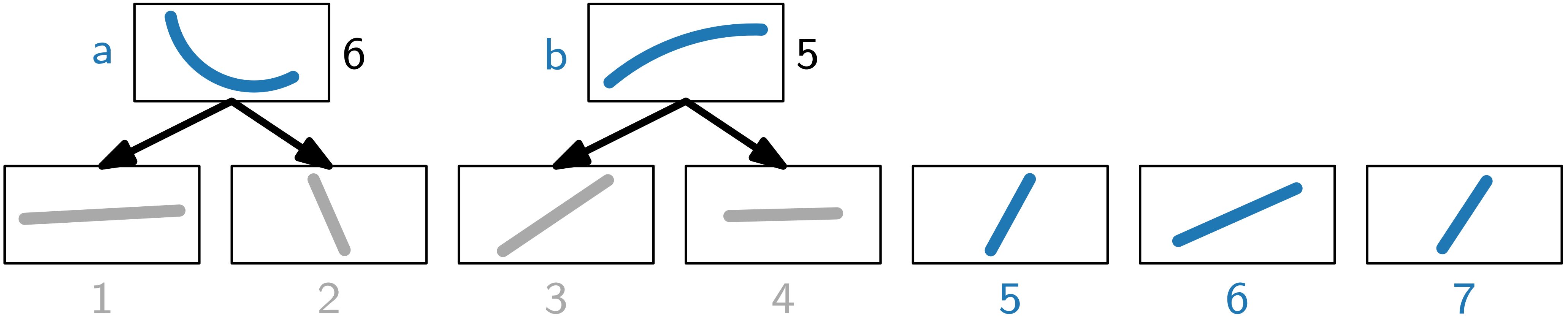




Recovering solutions

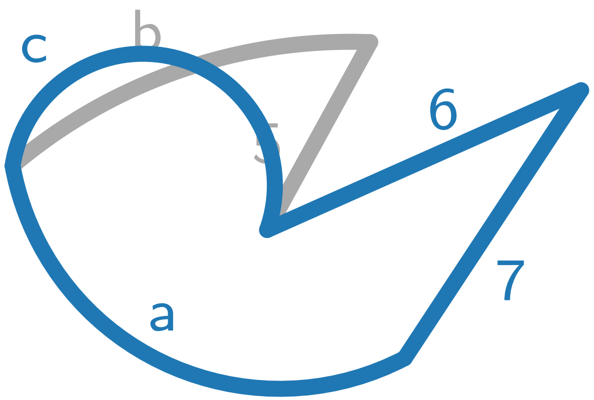
Run once, obtain all solutions

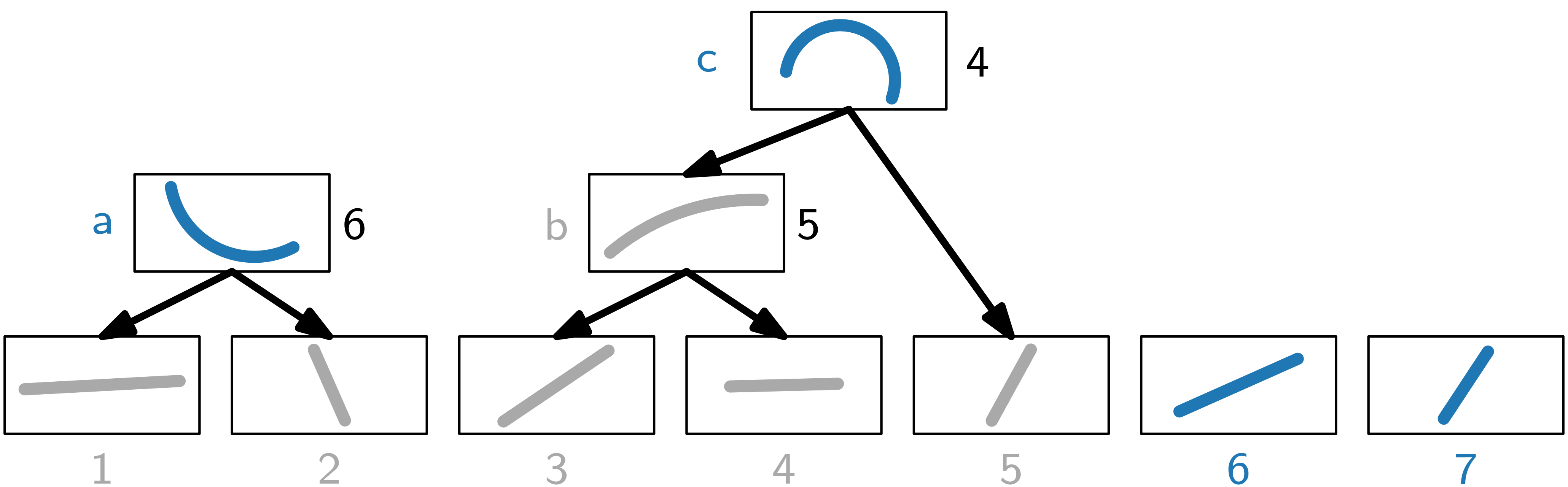




Recovering solutions

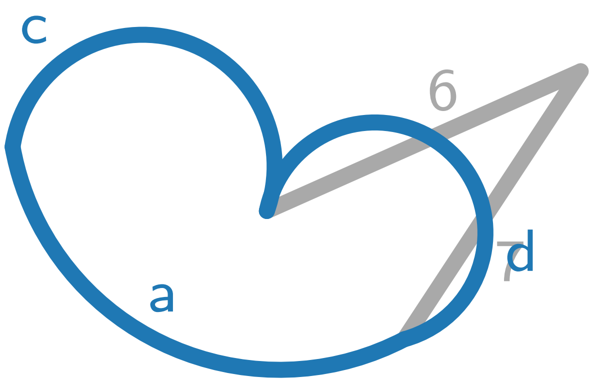
Run once, obtain all solutions

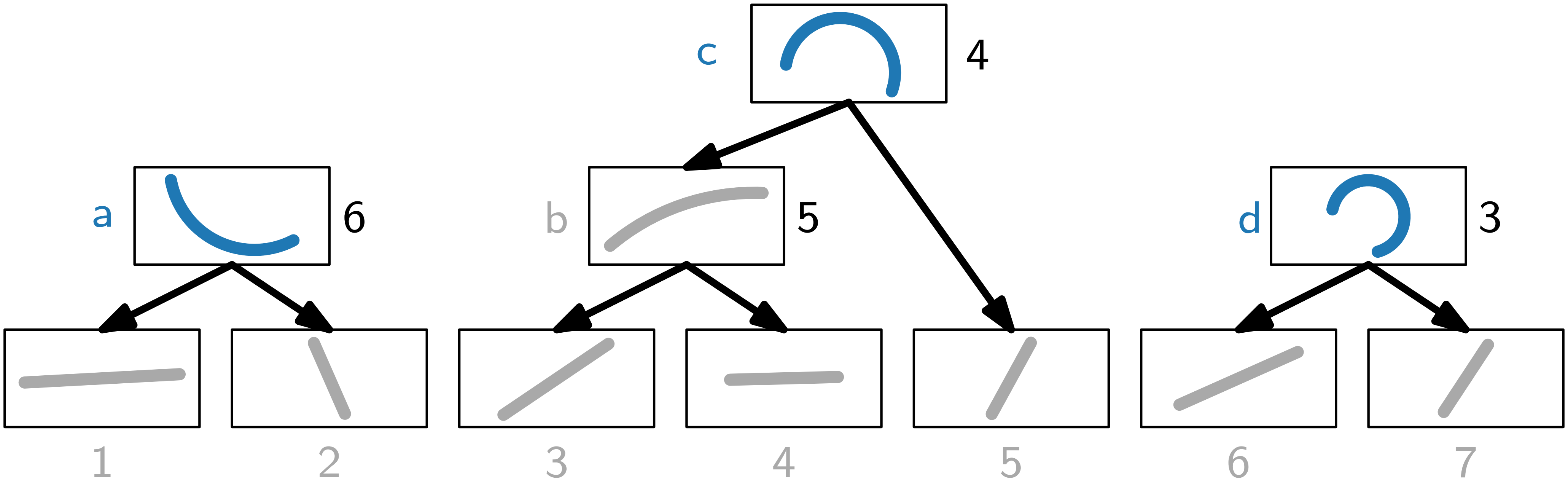




Recovering solutions

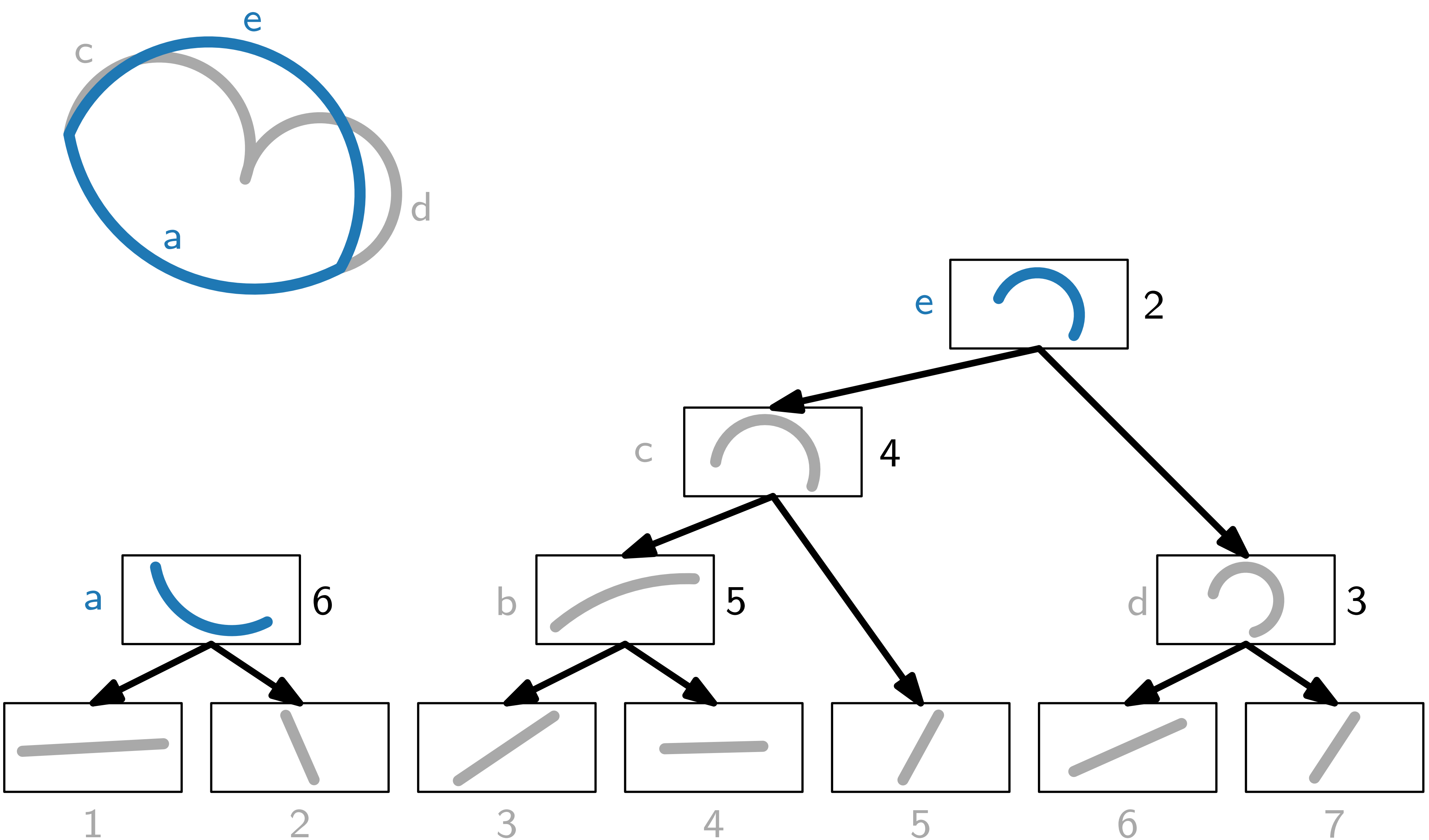
Run once, obtain all solutions





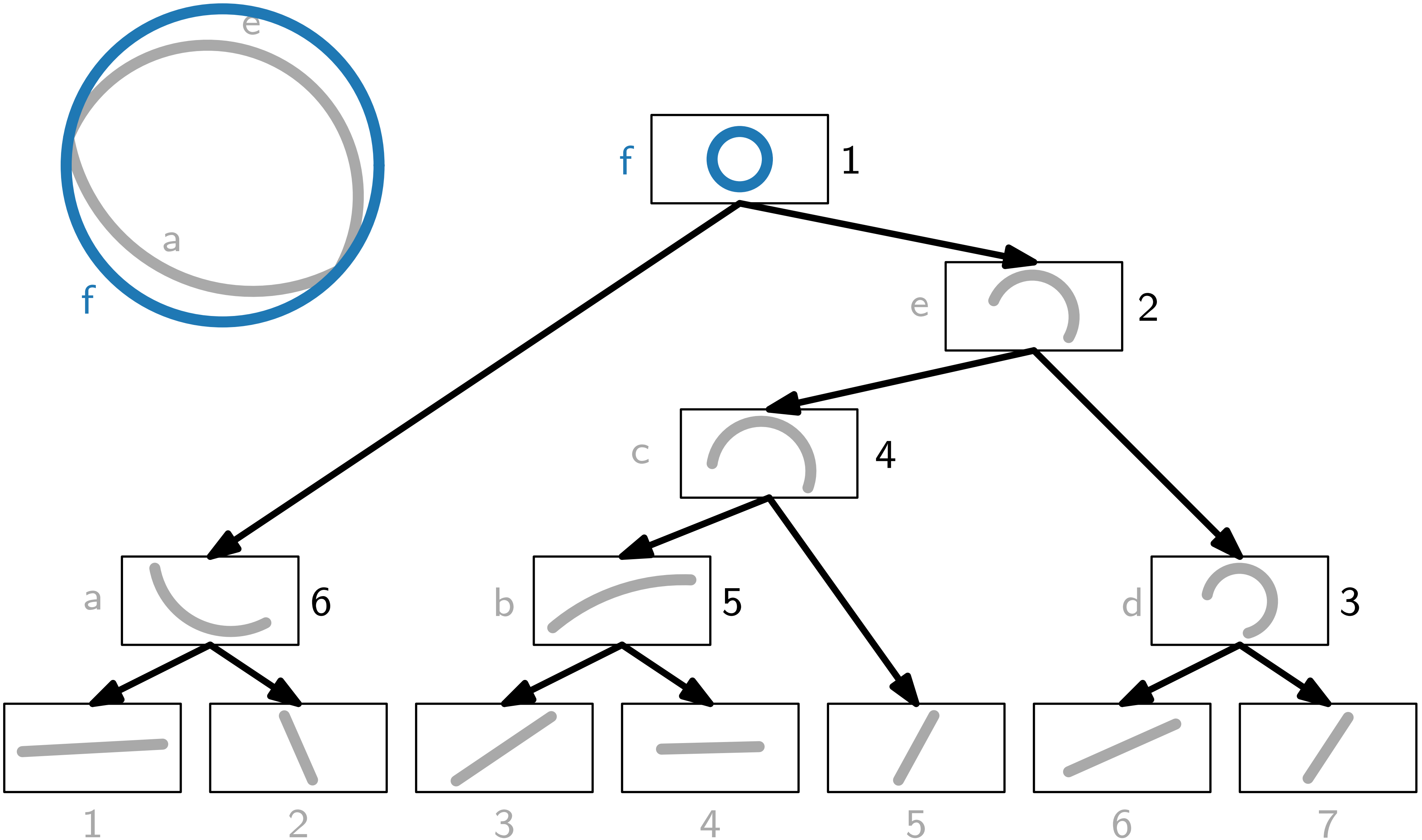
Recovering solutions

Run once, obtain all solutions



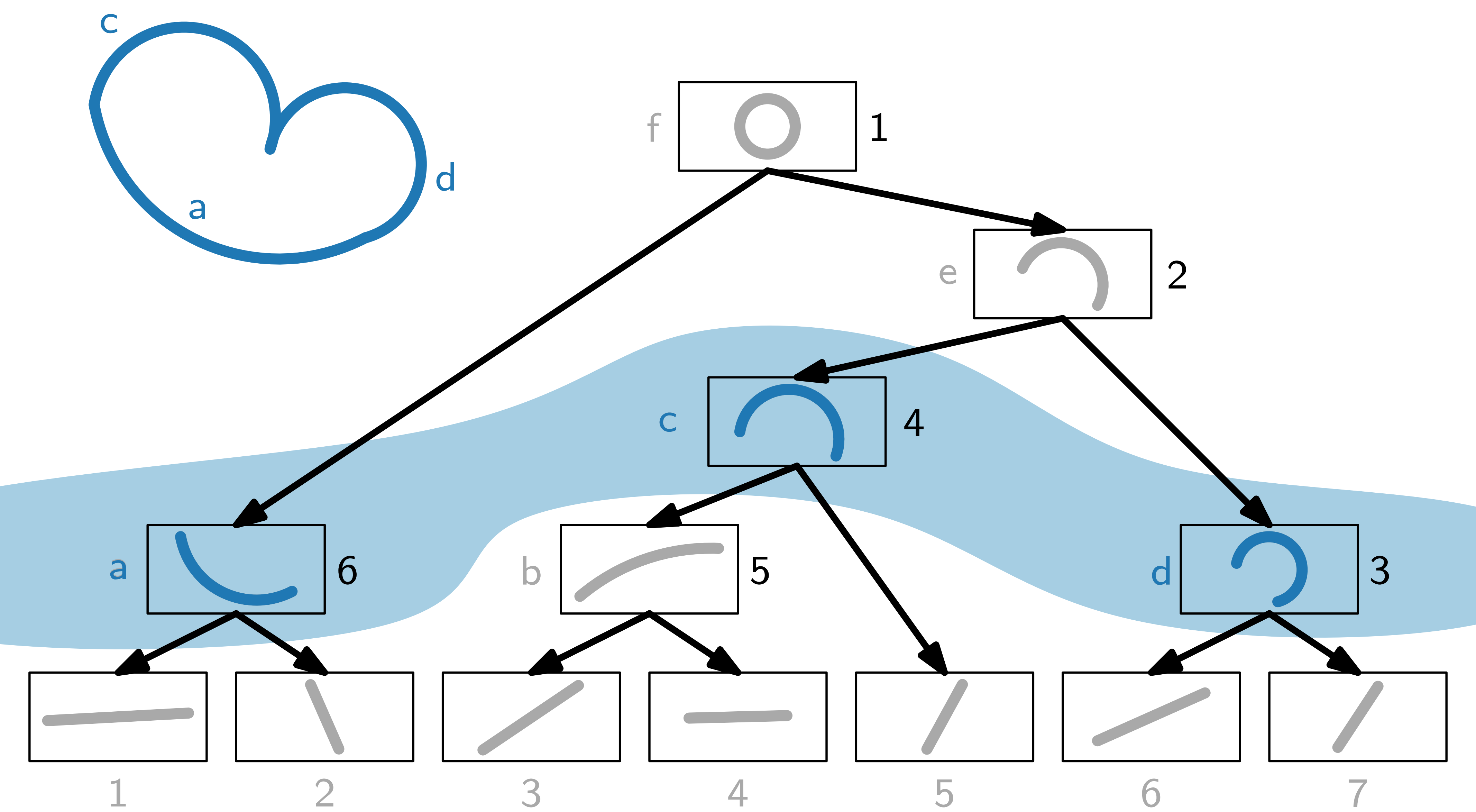
Recovering solutions

Run once, obtain all solutions



Recovering solutions

Run once, obtain all solutions

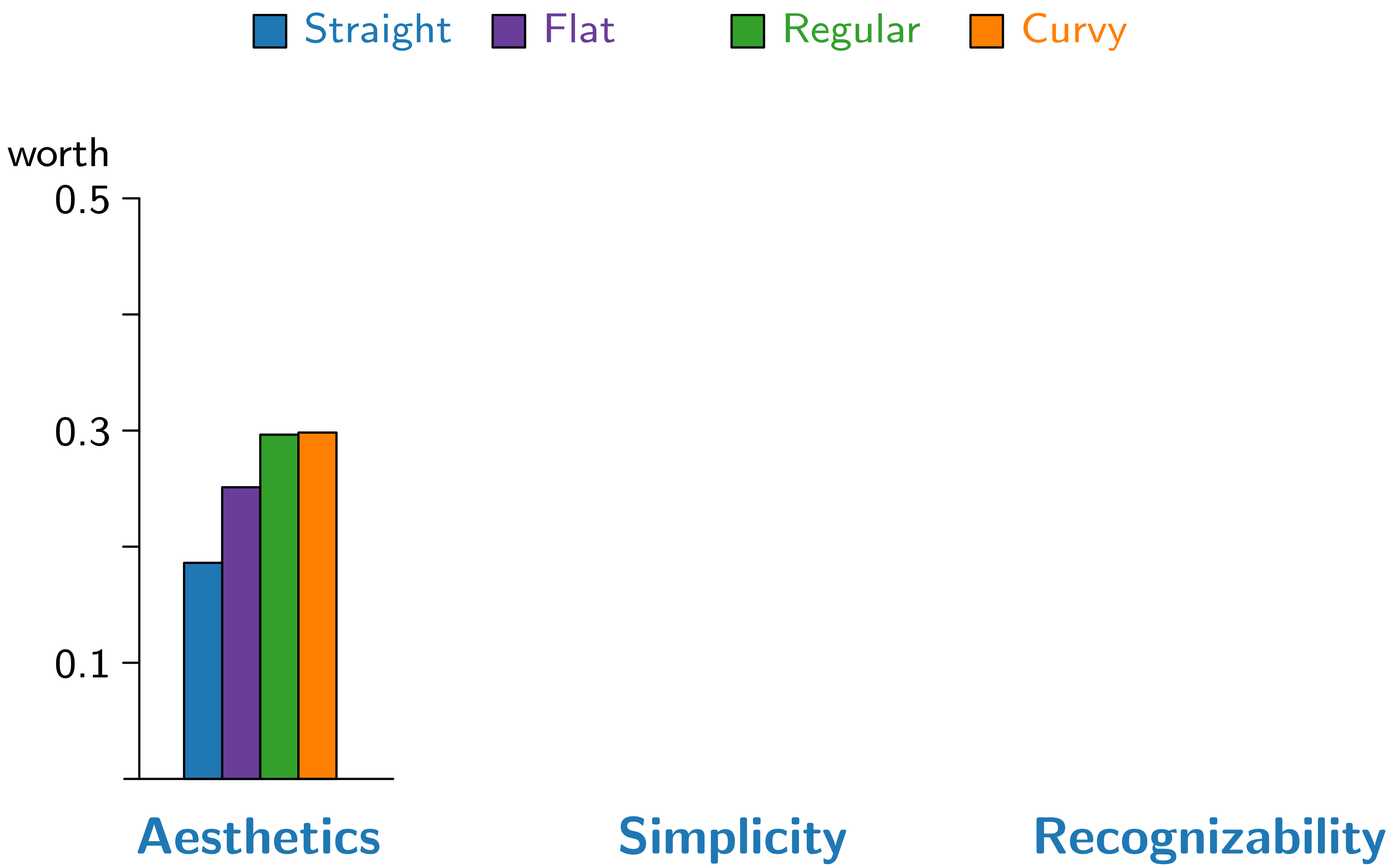


Lines vs arcs

Straight Flat Regular Curvy

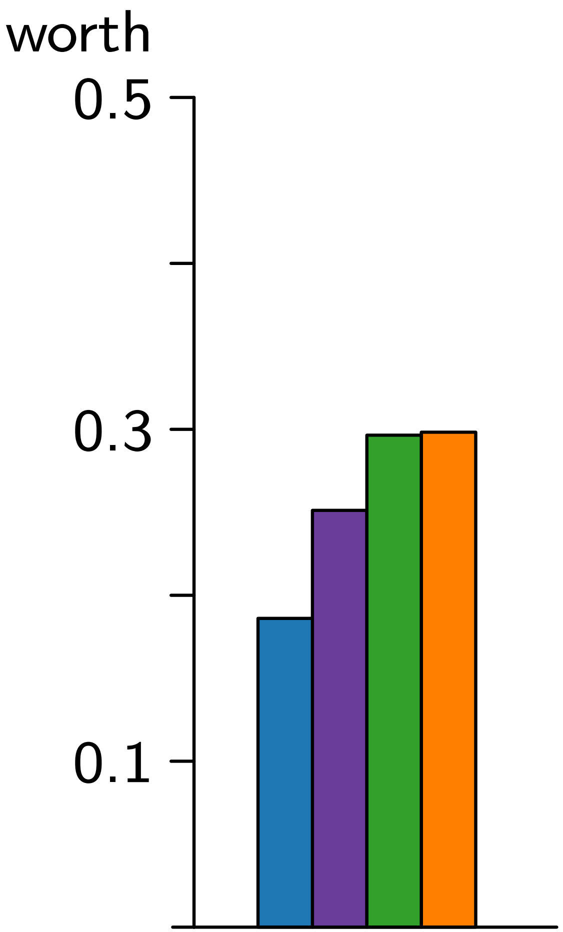
Aesthetics Simplicity Recognizability

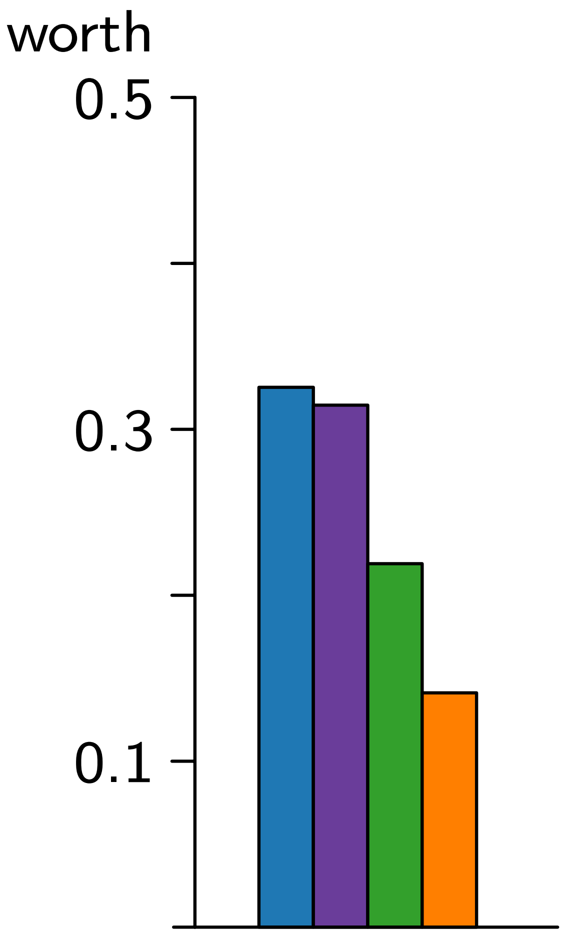
Lines vs arcs



Lines vs arcs

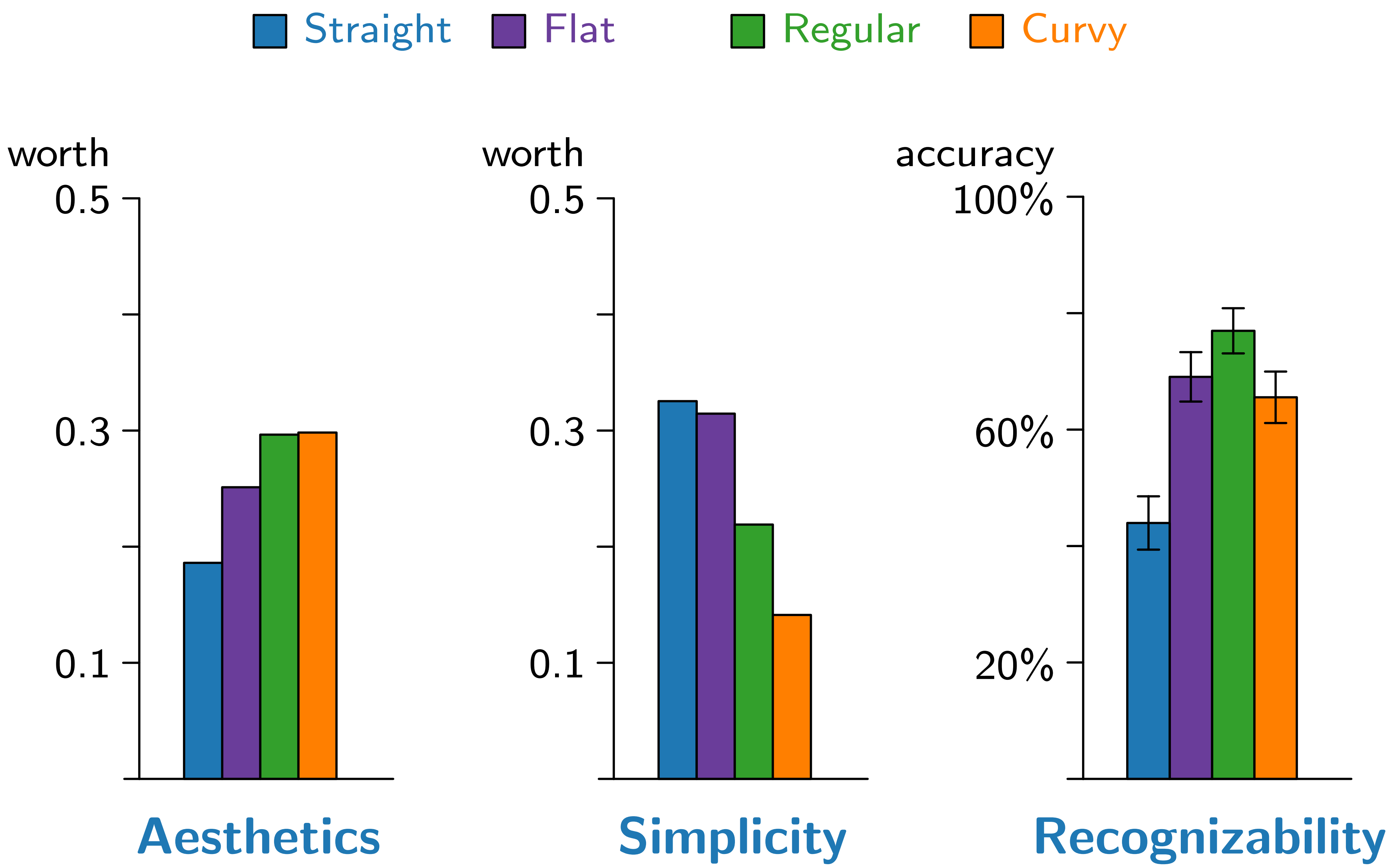
Straight Flat Regular Curvy





Aesthetics **Simplicity** **Recognizability**

Lines vs arcs



"Nongeographic" schematization

What happens if we get rid of all geography?

Problem.

Draw a graph G with **low complexity**

Graph complexity

Complexity of a graph $G = (V, E)$

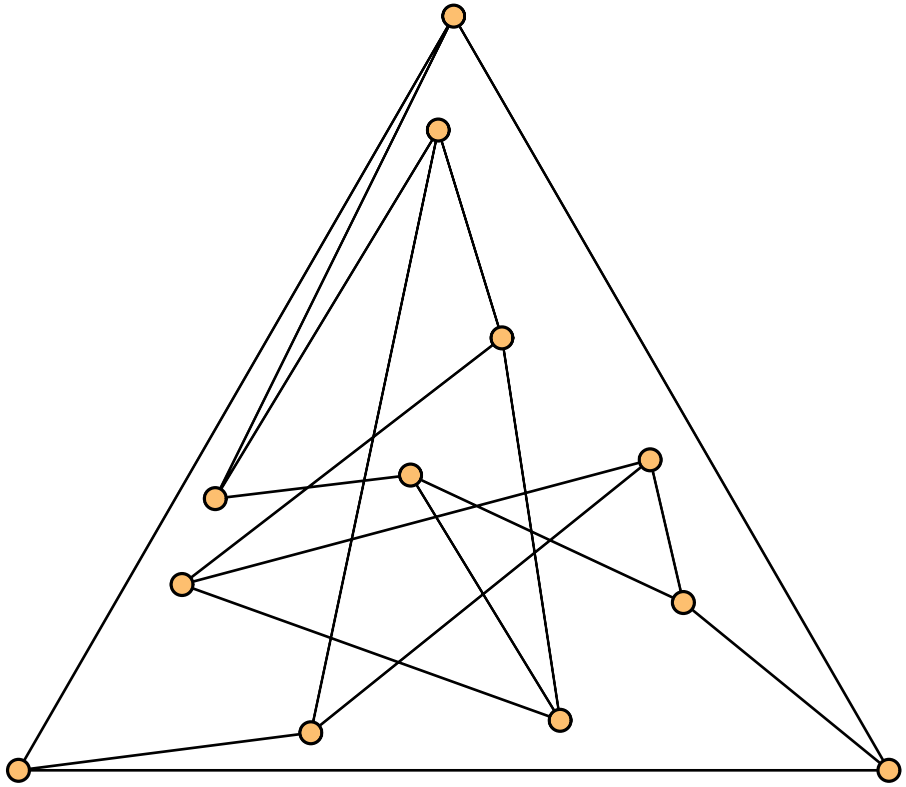
Usually $|V|$, $|E|$, etc.

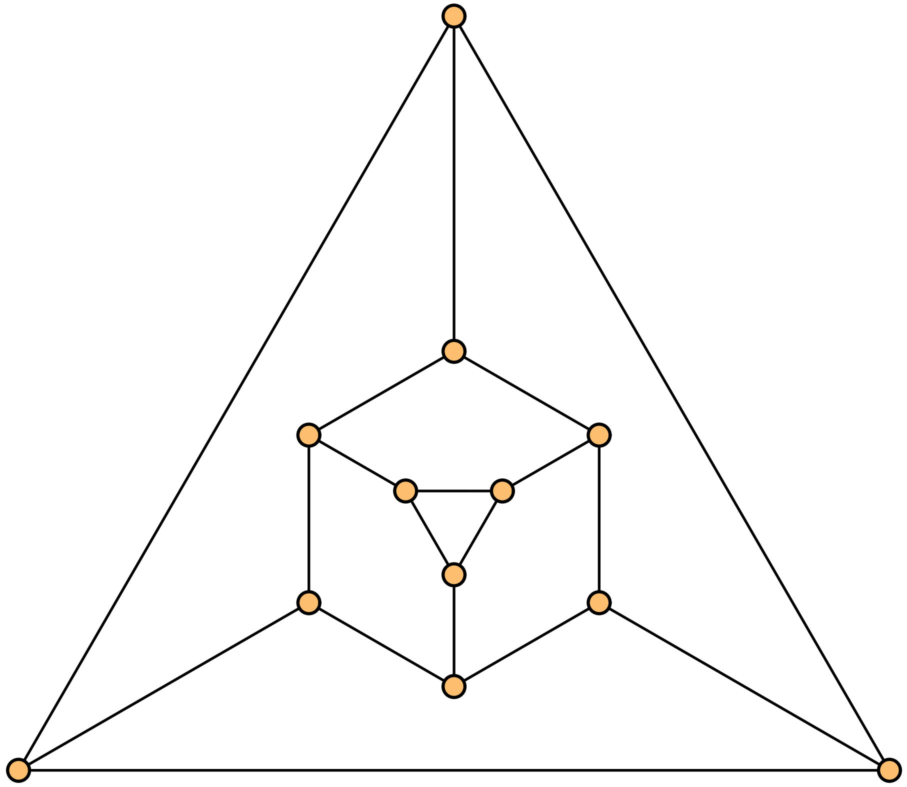
Graph complexity

Complexity of a graph $G = (V, E)$

Usually $|V|$, $|E|$, etc.

Says nothing about how complex a drawing is

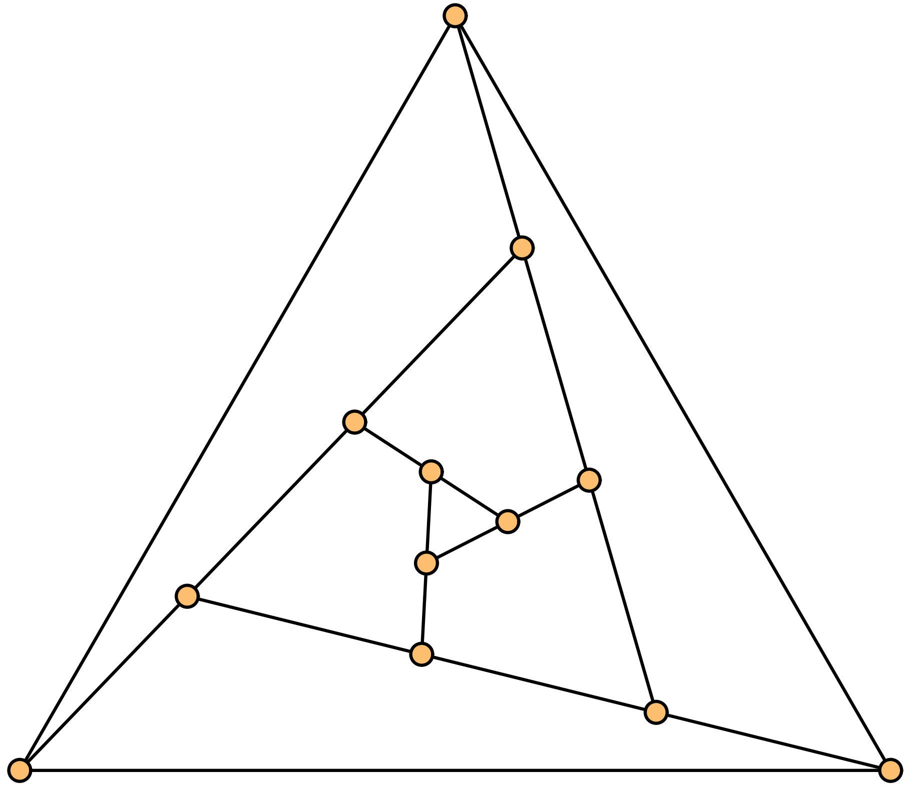




Visual complexity

Planar graphs

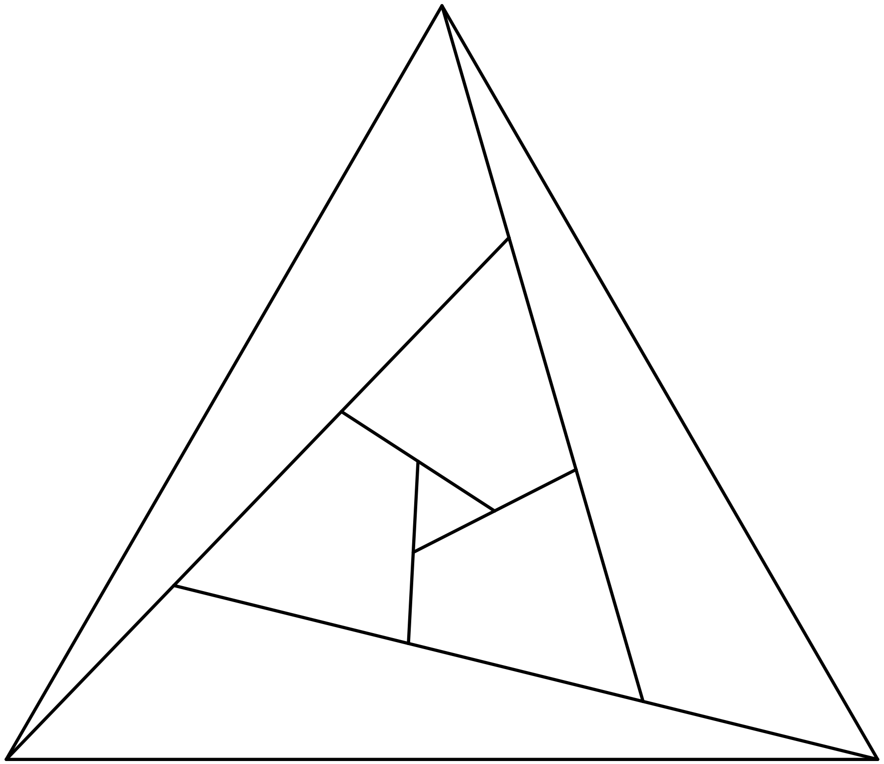
Number of **geometric objects** for drawing



Visual complexity

Planar graphs

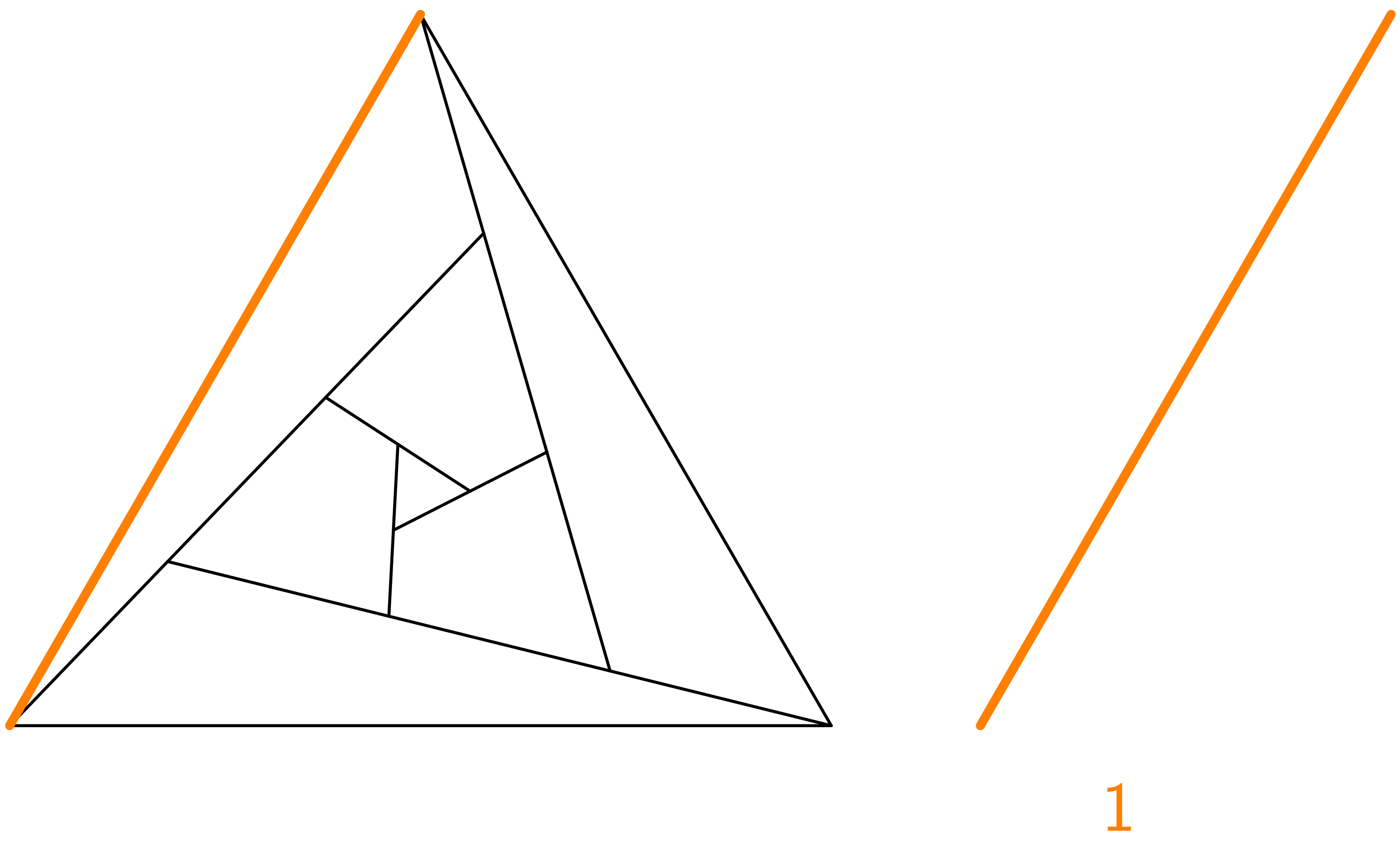
Number of **geometric objects** for drawing



Visual complexity

Planar graphs

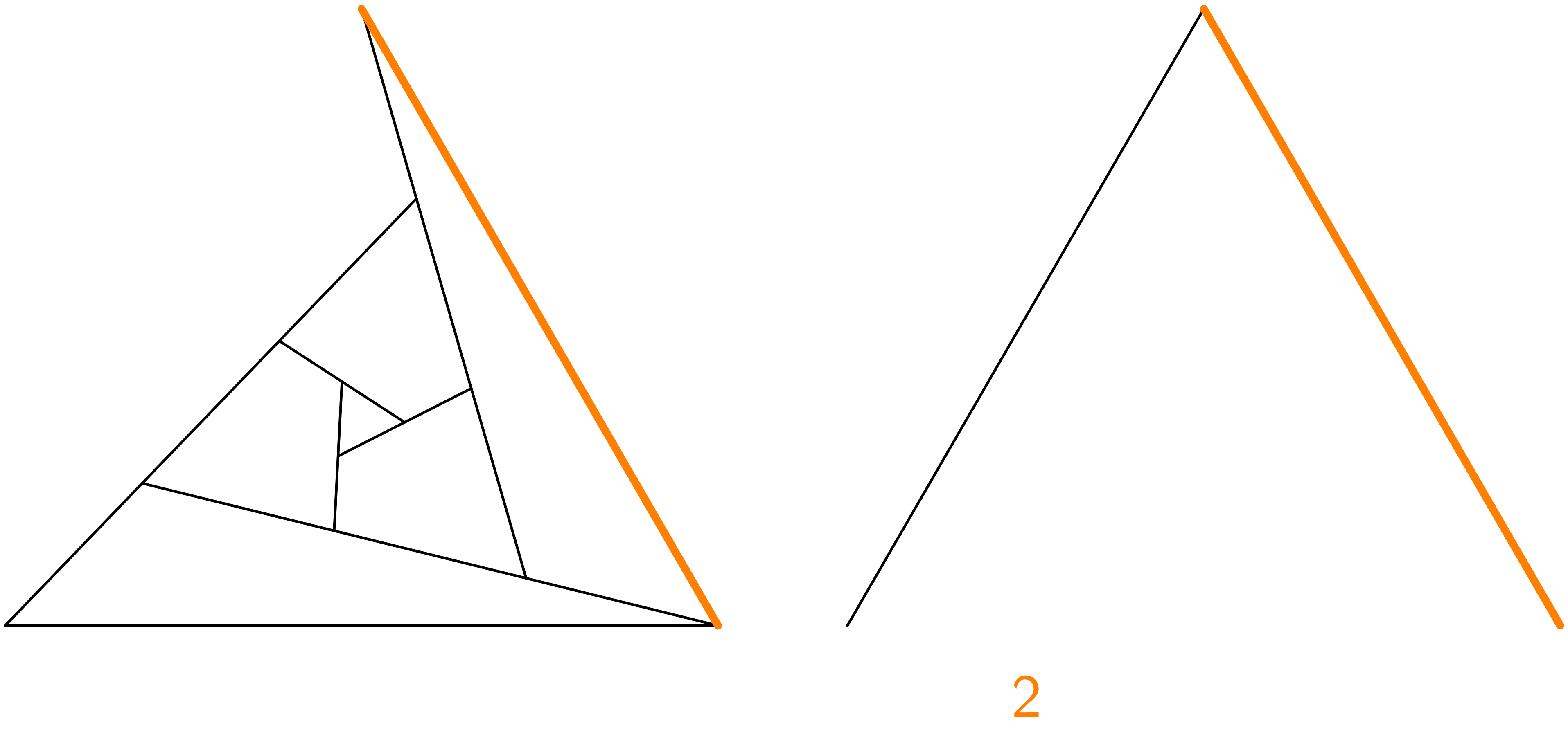
Number of **geometric objects** for drawing



Visual complexity

Planar graphs

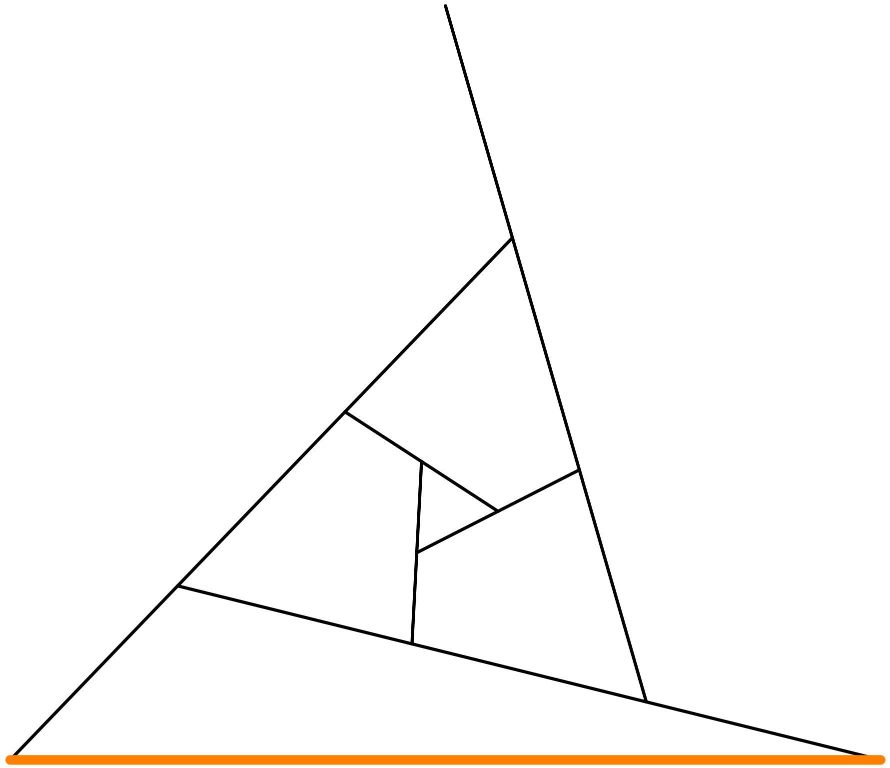
Number of **geometric objects** for drawing

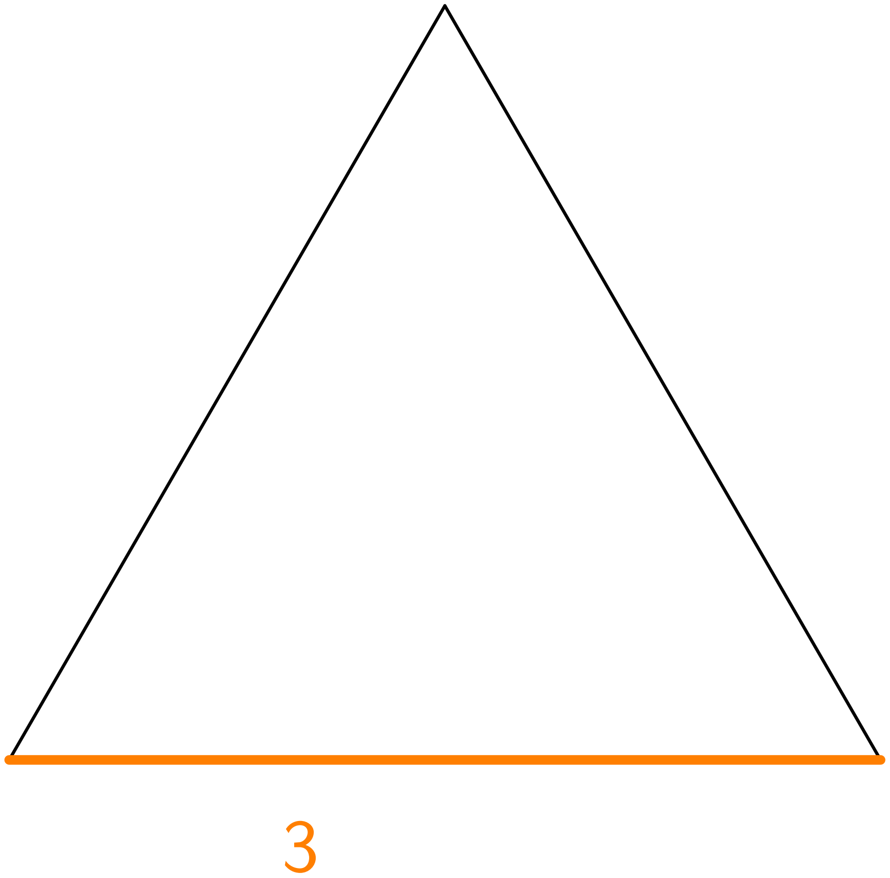


Visual complexity

Planar graphs

Number of **geometric objects** for drawing

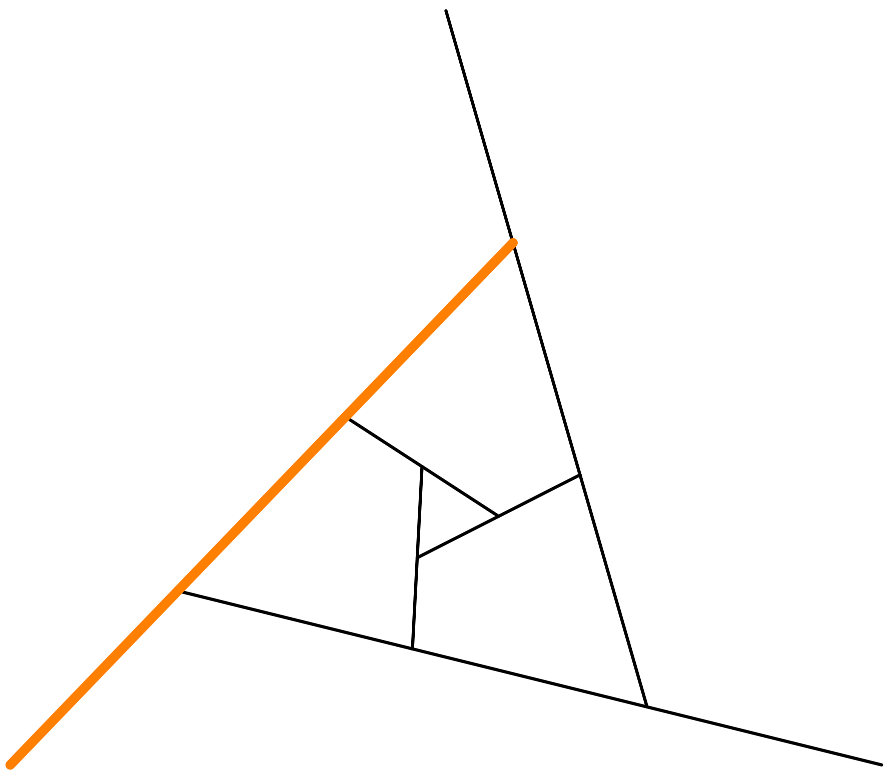


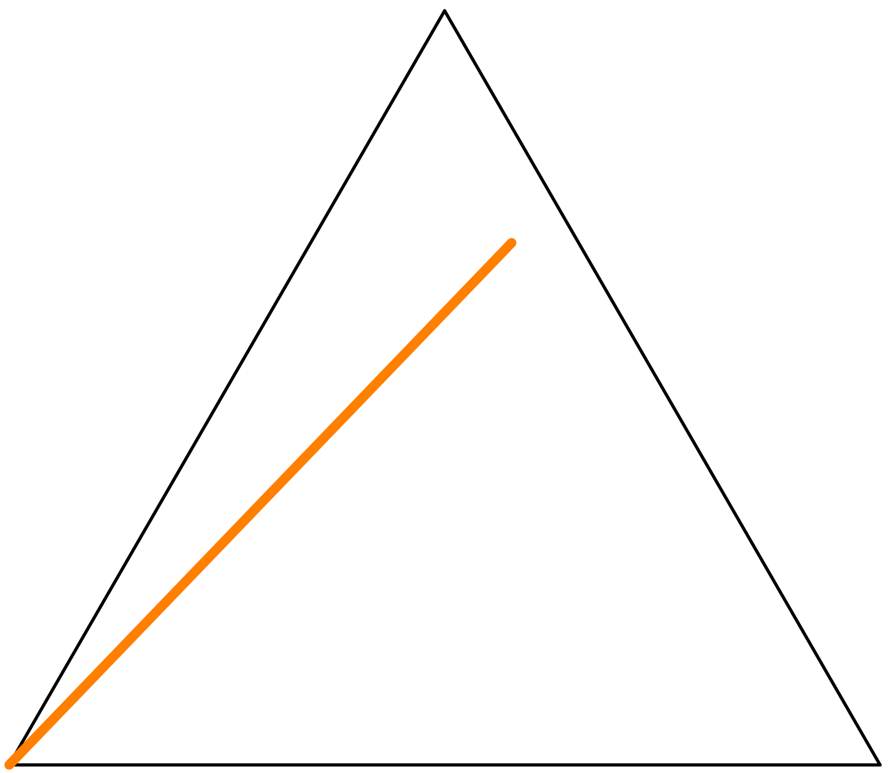


Visual complexity

Planar graphs

Number of **geometric objects** for drawing



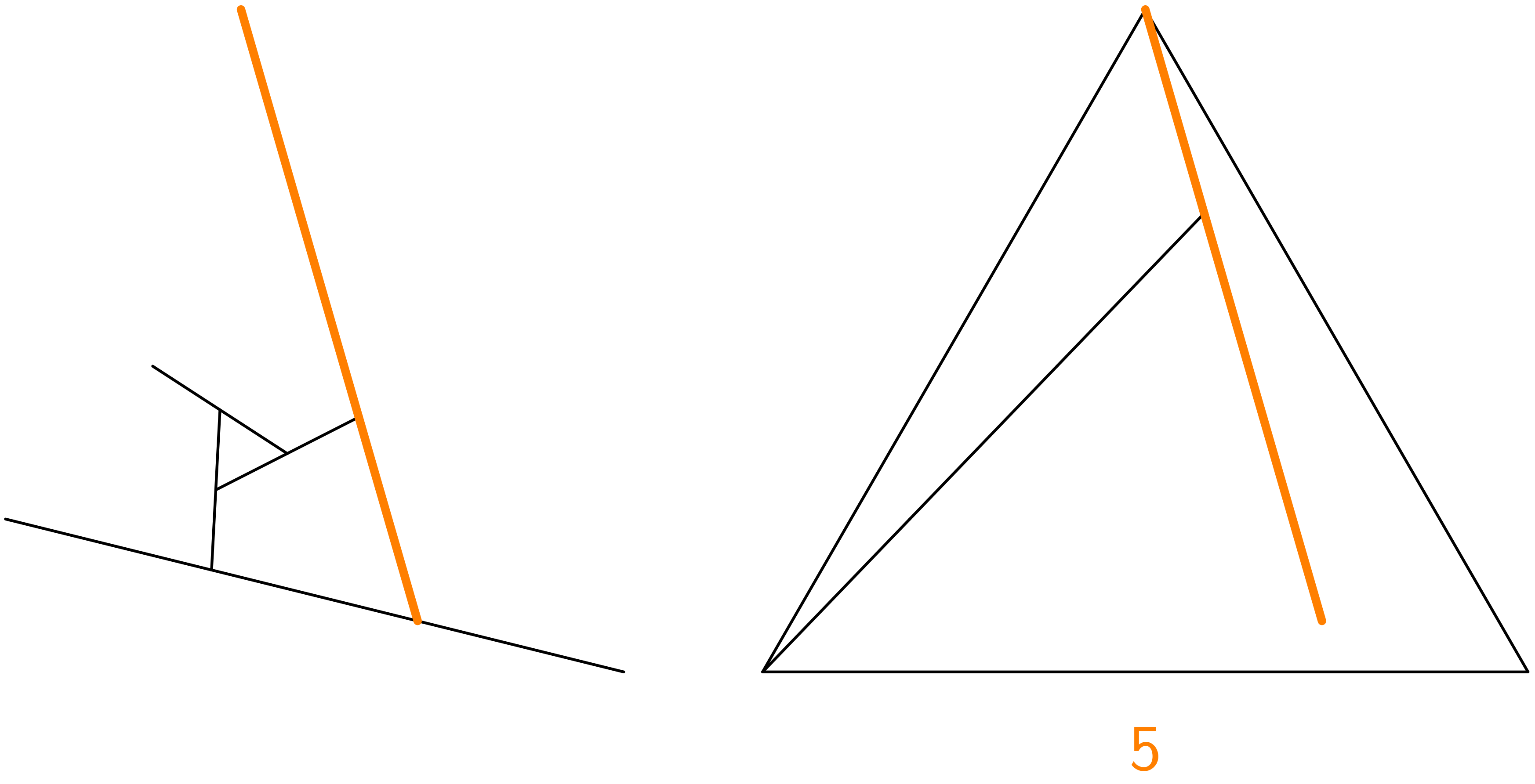


4

Visual complexity

Planar graphs

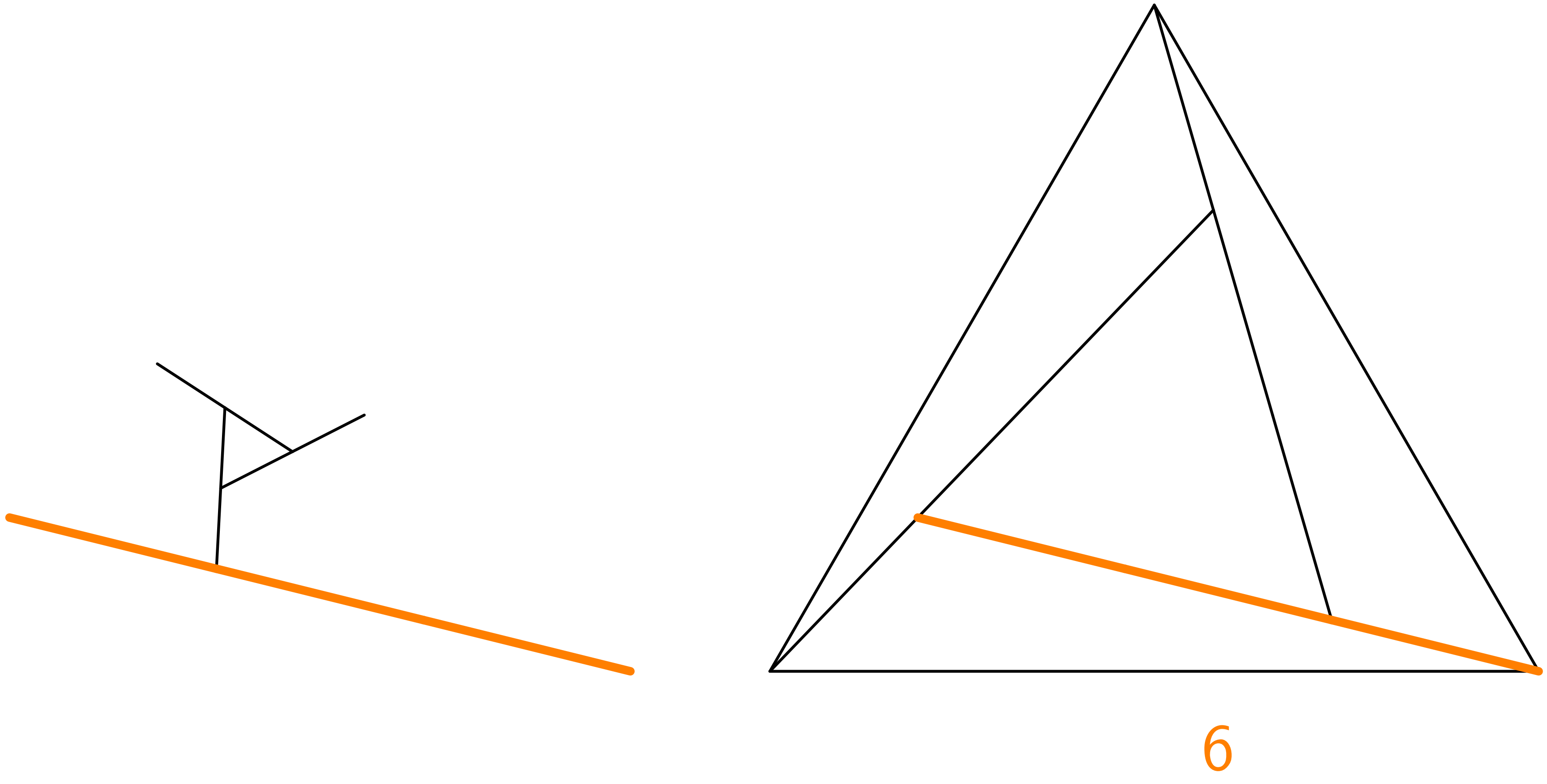
Number of **geometric objects** for drawing



Visual complexity

Planar graphs

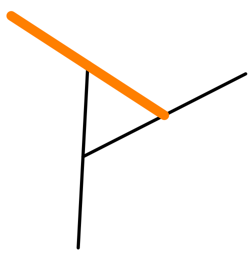
Number of **geometric objects** for drawing

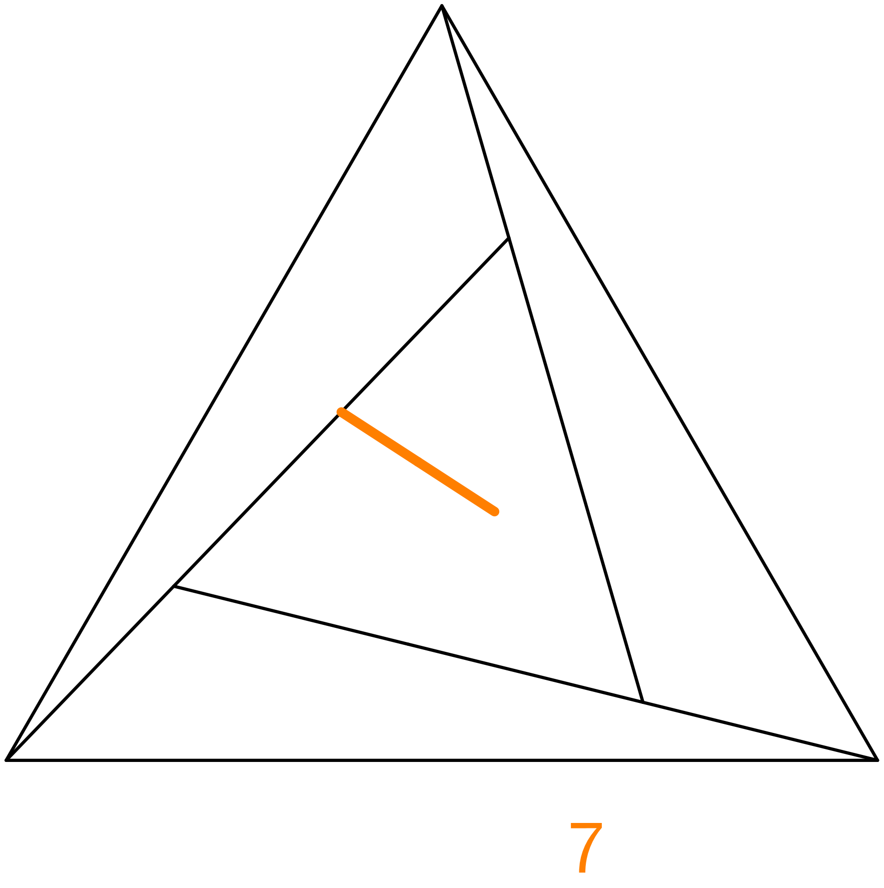


Visual complexity

Planar graphs

Number of **geometric objects** for drawing

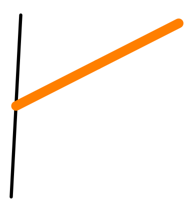


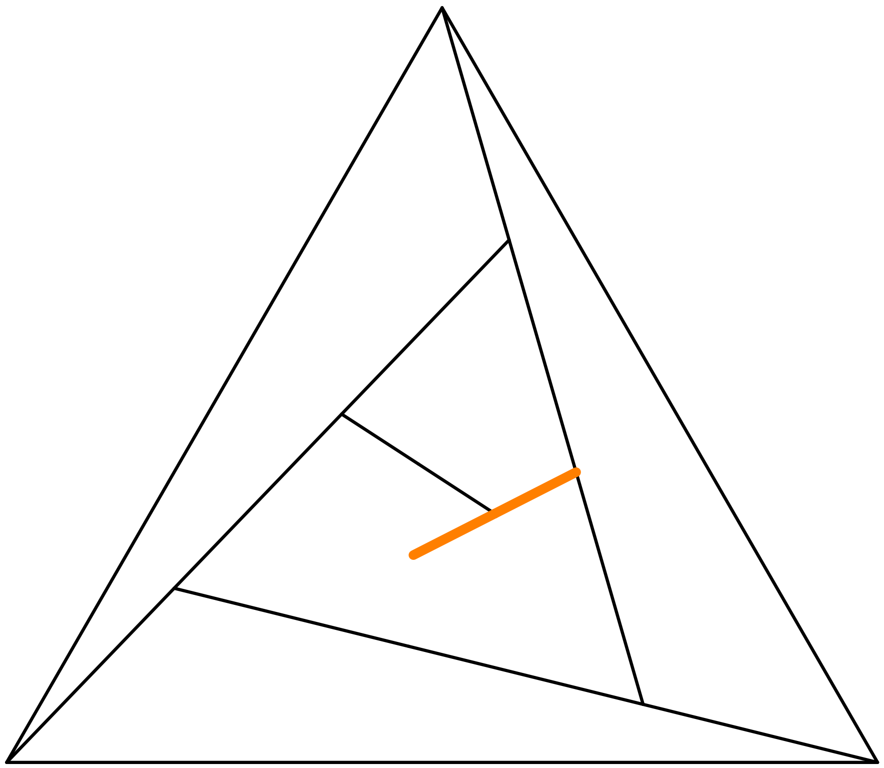


Visual complexity

Planar graphs

Number of **geometric objects** for drawing



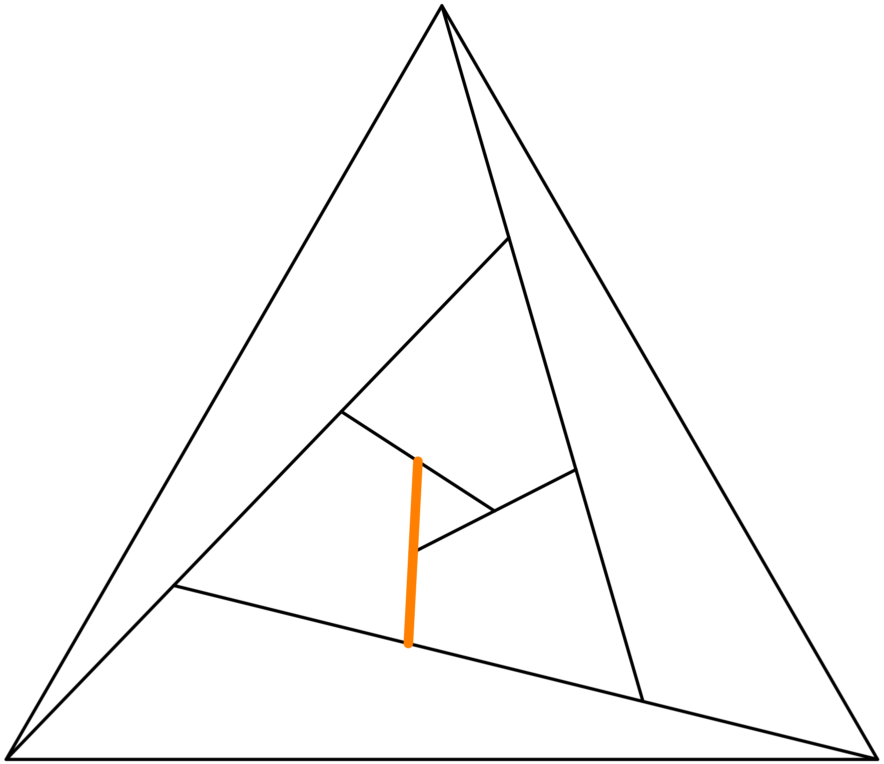


8

Visual complexity

Planar graphs

Number of **geometric objects** for drawing

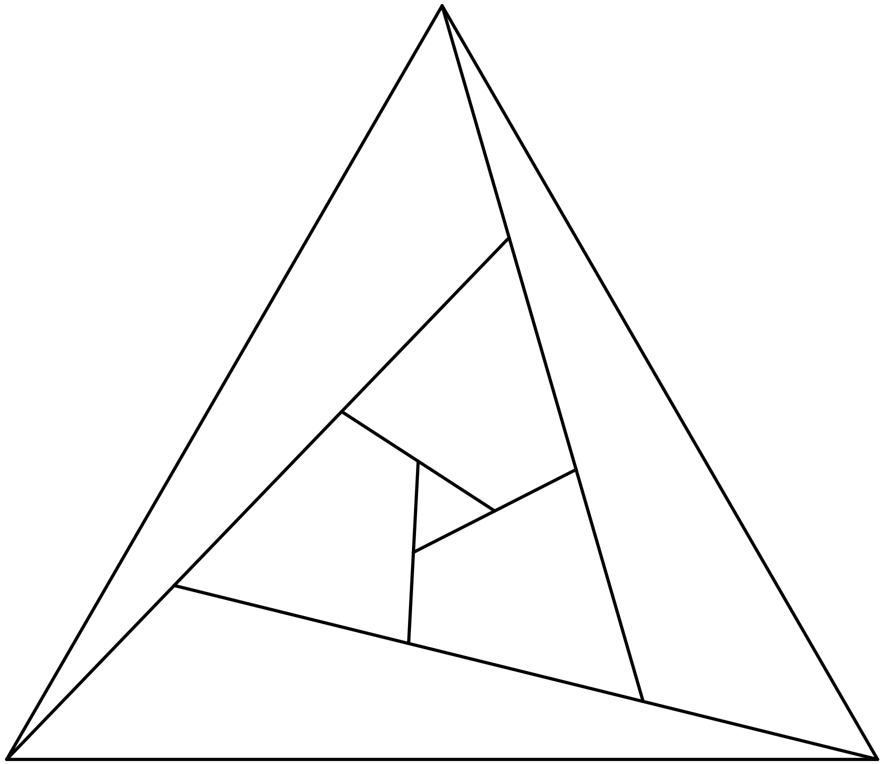


9

Visual complexity

Planar graphs

Number of **geometric objects** for drawing



9 line segments for 18 edges

Known results

	Class	Lower	Upper	
Segments	Tree	$K/2$	$K/2$	[Durocher et al, 2013]
	2- and 3-trees	$2V$	$2V$	[Dujmović et al, 2007]
	3-connected	$2V$	$5V/2$	[Dujmović et al, 2007]
	Triangulation	$2V$	$7V/3$	[Durocher, Mondal, 2014]
	Planar	$2V$	$16V/3 - E$	[Durocher, Mondal, 2014]

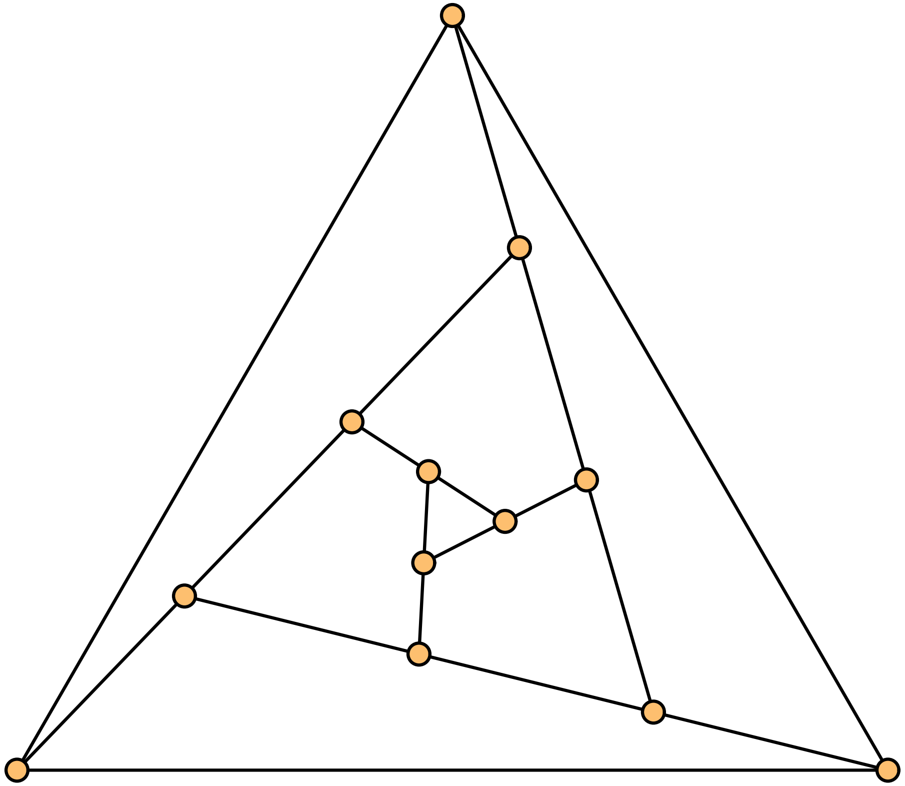
Known results

	Class	Lower	Upper	
Segments	Tree	$K/2$	$K/2$	[Durocher et al, 2013]
	2- and 3-trees	$2V$	$2V$	[Dujmović et al, 2007]
	3-connected	$2V$	$5V/2$	[Dujmović et al, 2007]
	Triangulation	$2V$	$7V/3$	[Durocher, Mondal, 2014]
	Planar	$2V$	$16V/3 - E$	[Durocher, Mondal, 2014]
Circ. arcs	3-trees	$E/6$	$11E/18$	[Schulz, 2013]
	3-connected	$E/6$	$2E/3$	[Schulz, 2013]

The remainder of this talk

Line-segment drawings

Planar cubic 3-connected graphs



The remainder of this talk

Line-segment drawings

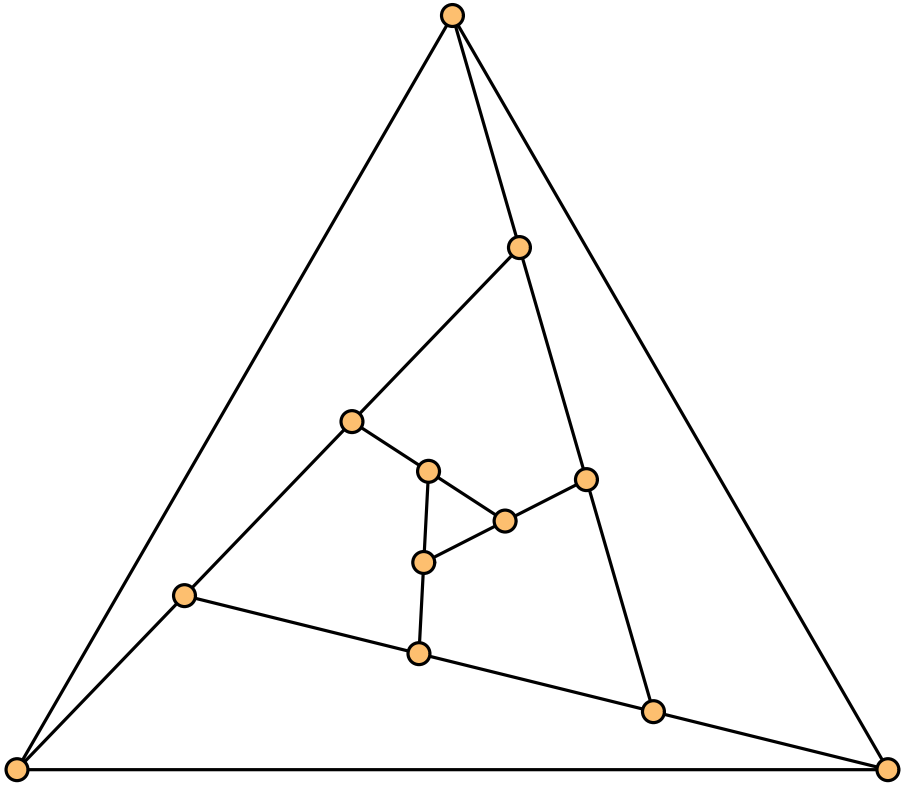
Planar cubic 3-connected graphs

Two new algorithms

$n/2 + 3$ segments

[Mondal et al, 2013]

Resolve flaw & improved



The remainder of this talk

Line-segment drawings

Planar cubic 3-connected graphs

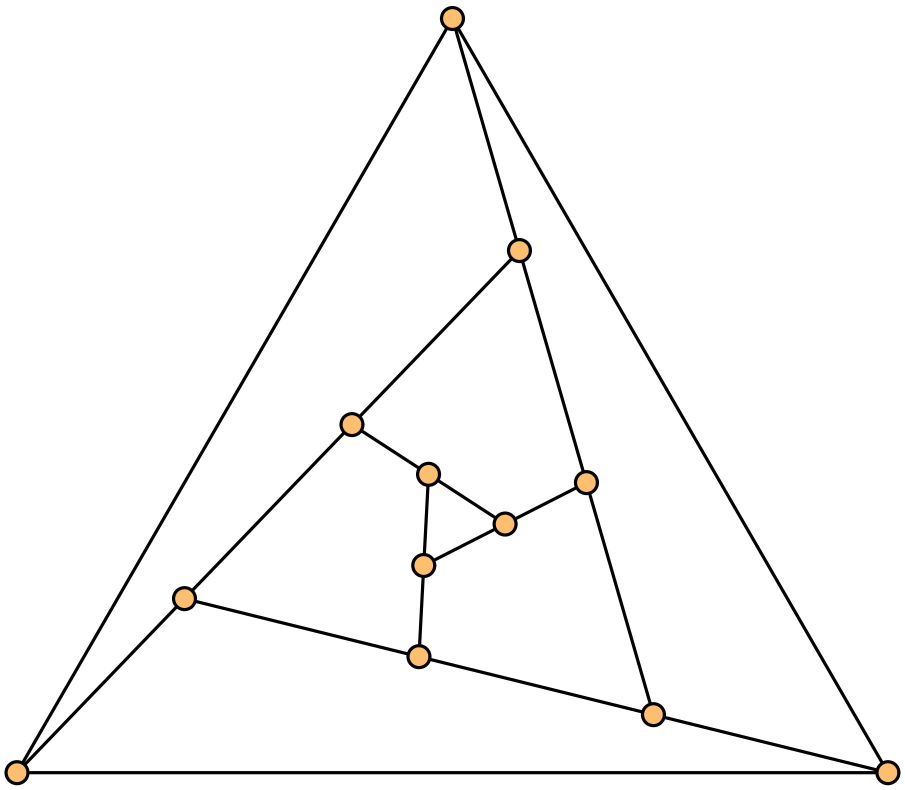
Two new algorithms

$n/2 + 3$ segments

[Mondal et al, 2013]

Resolve flaw & improved

Experimental comparison



Deconstruction algorithm

Deconstruction algorithm

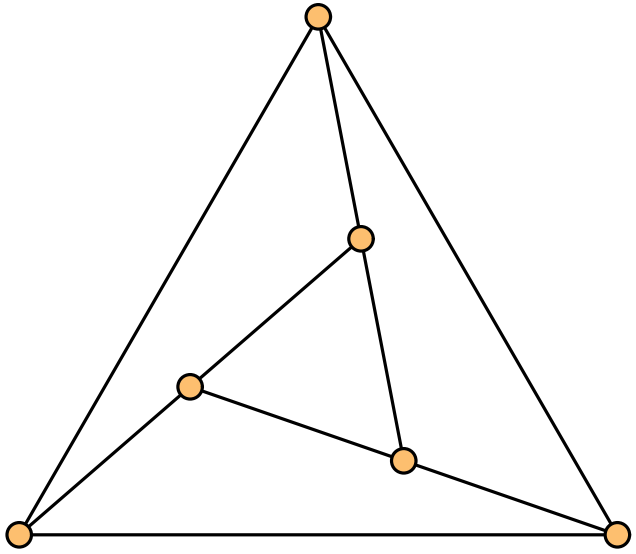
Theorem.

Every graph can be constructed

from the triangular prism

with **insertions**

maintaining a given outer face.



Deconstruction algorithm

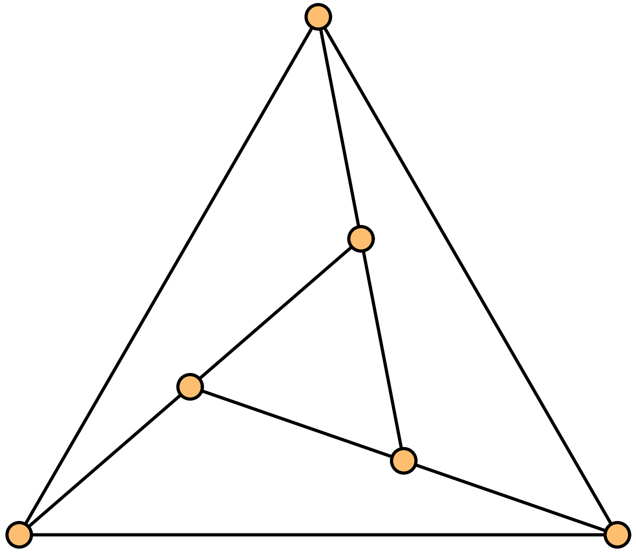
Theorem.

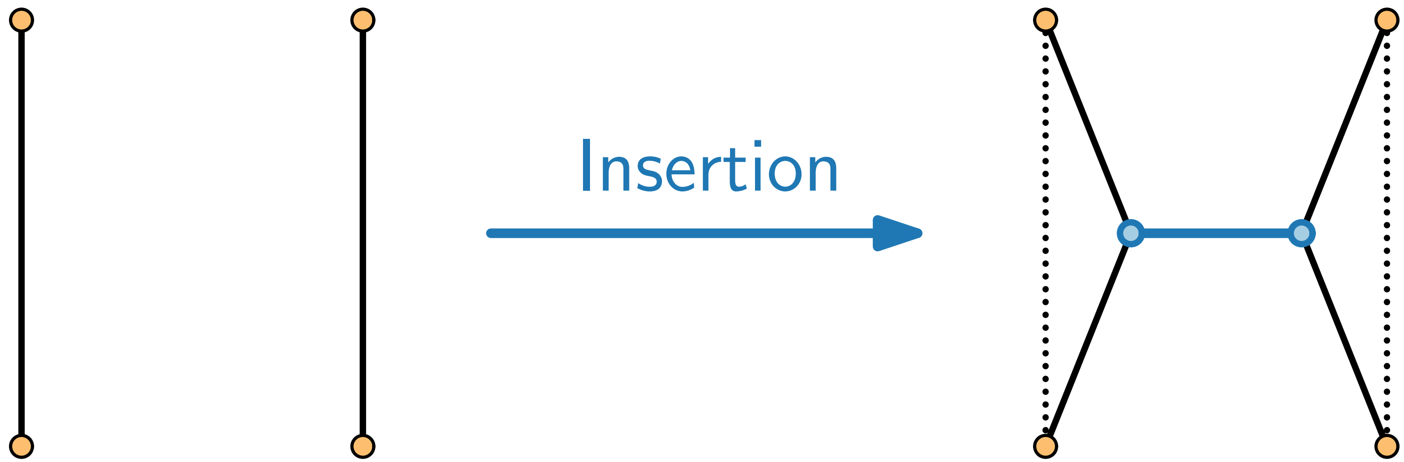
Every graph can be constructed

from the triangular prism

with **insertions**

maintaining a given outer face.

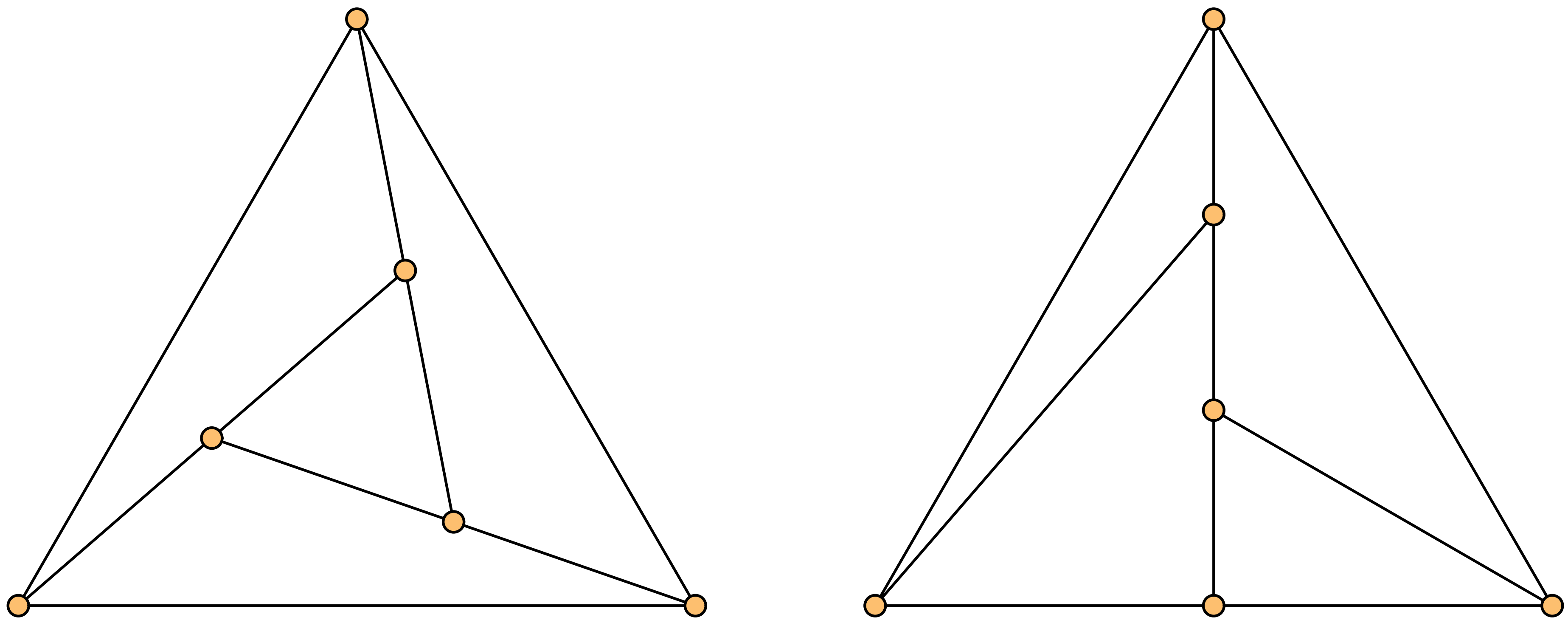




Deconstruction algorithm

Algorithm

1. Draw triangular prism

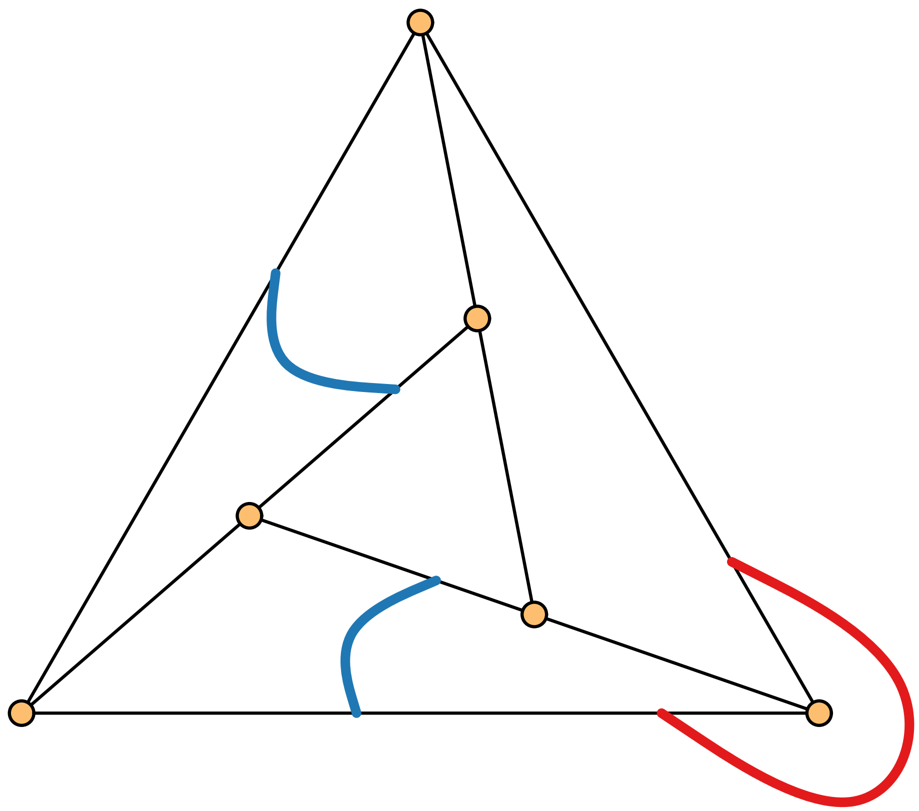


Deconstruction algorithm

Algorithm

1. Draw triangular prism

2. Construct graph, maintaining drawing



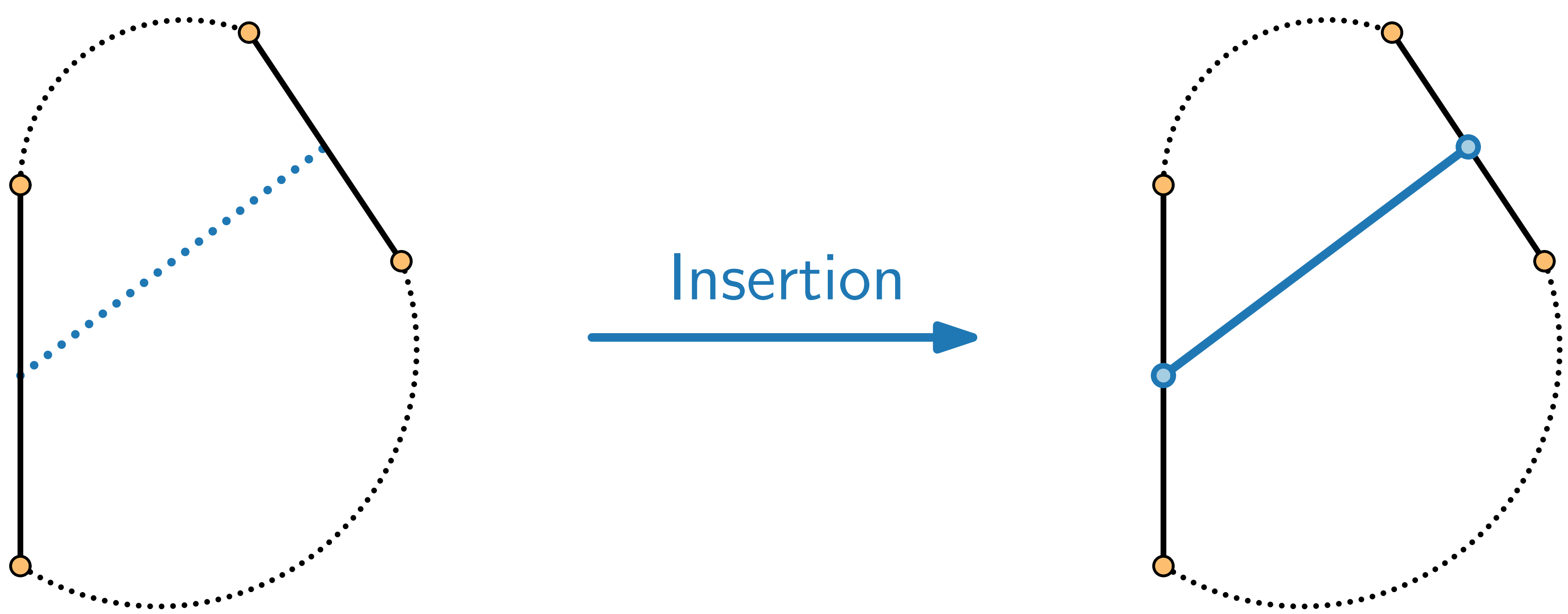
Inner faces are convex

No insertions on outer face

Deconstruction algorithm

Algorithm

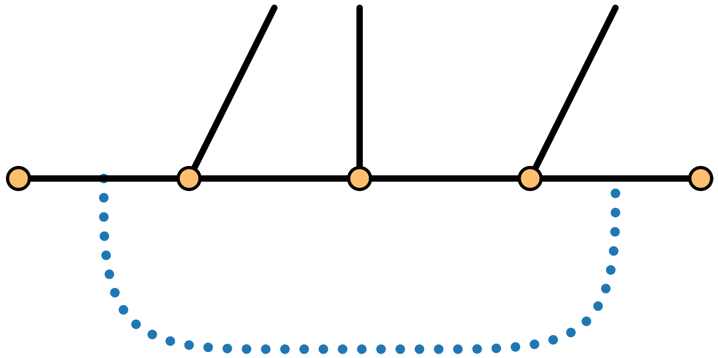
1. Draw triangular prism
2. Construct graph, maintaining drawing



Deconstruction algorithm

Algorithm

1. Draw triangular prism
2. Construct graph, maintaining drawing

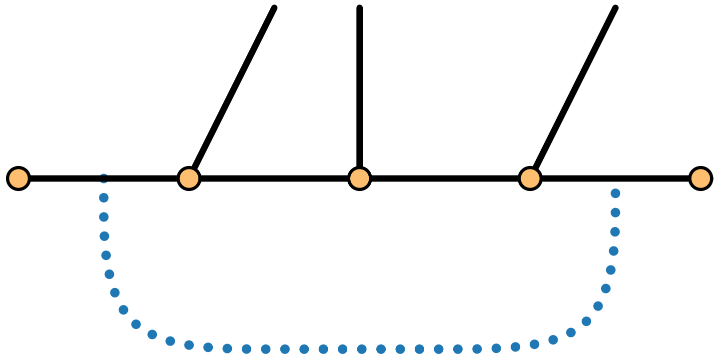


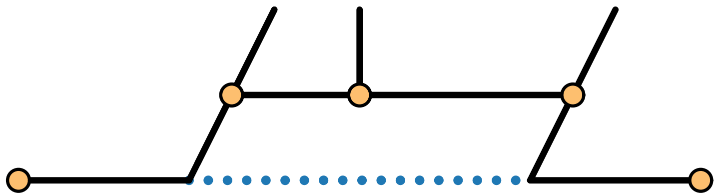
Deconstruction algorithm

Algorithm

1. Draw triangular prism

2. Construct graph, maintaining drawing

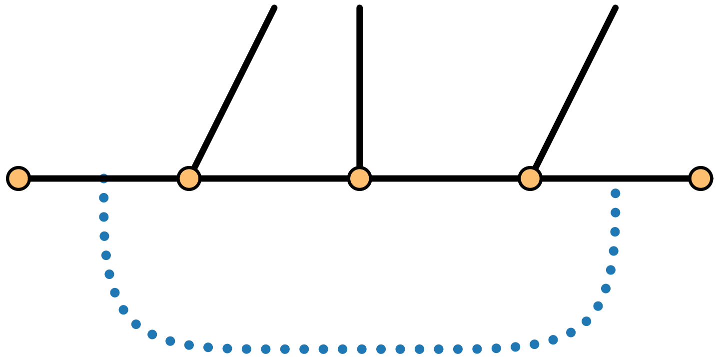




Deconstruction algorithm

Algorithm

1. Draw triangular prism
2. Construct graph, maintaining drawing



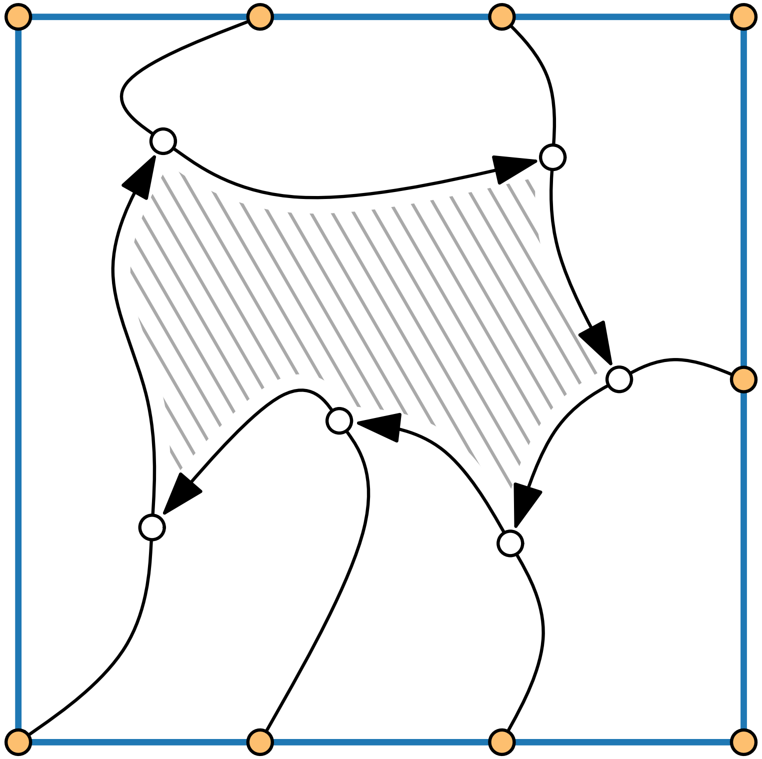
Windmill algorithm

Windmill algorithm

Algorithm

Pre: cycle C drawn convex

Post: inside of C drawn

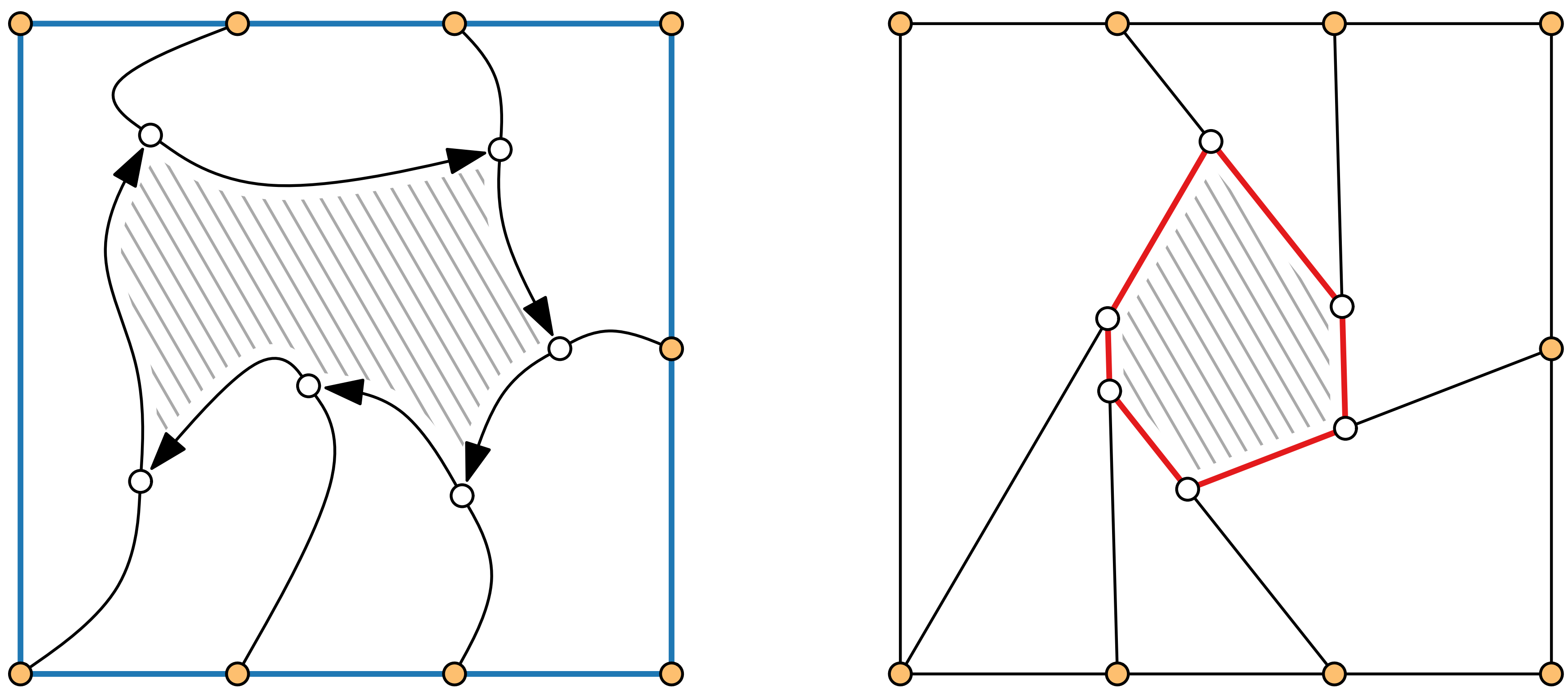


Windmill algorithm

Algorithm

Pre: cycle C drawn convex

Post: inside of C drawn

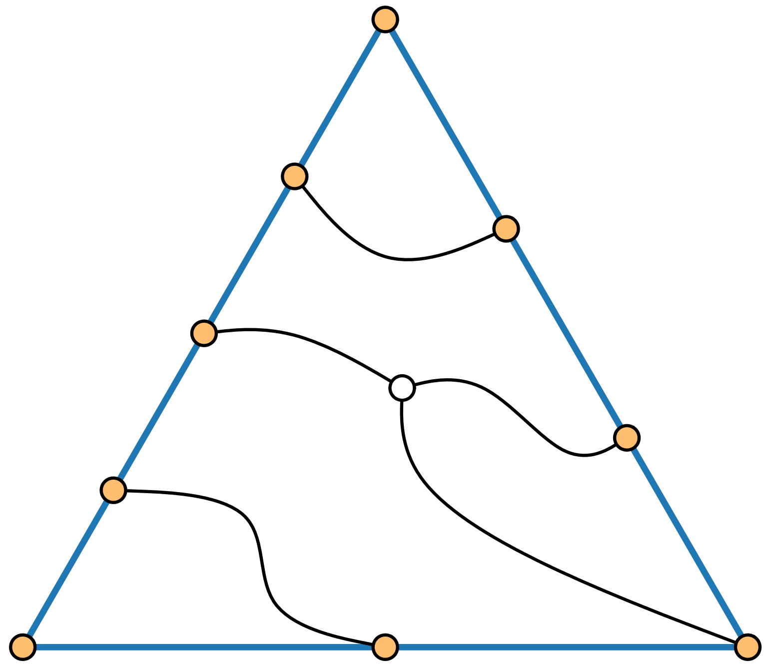


Windmill algorithm

Algorithm

Pre: cycle C drawn convex

Post: inside of C drawn

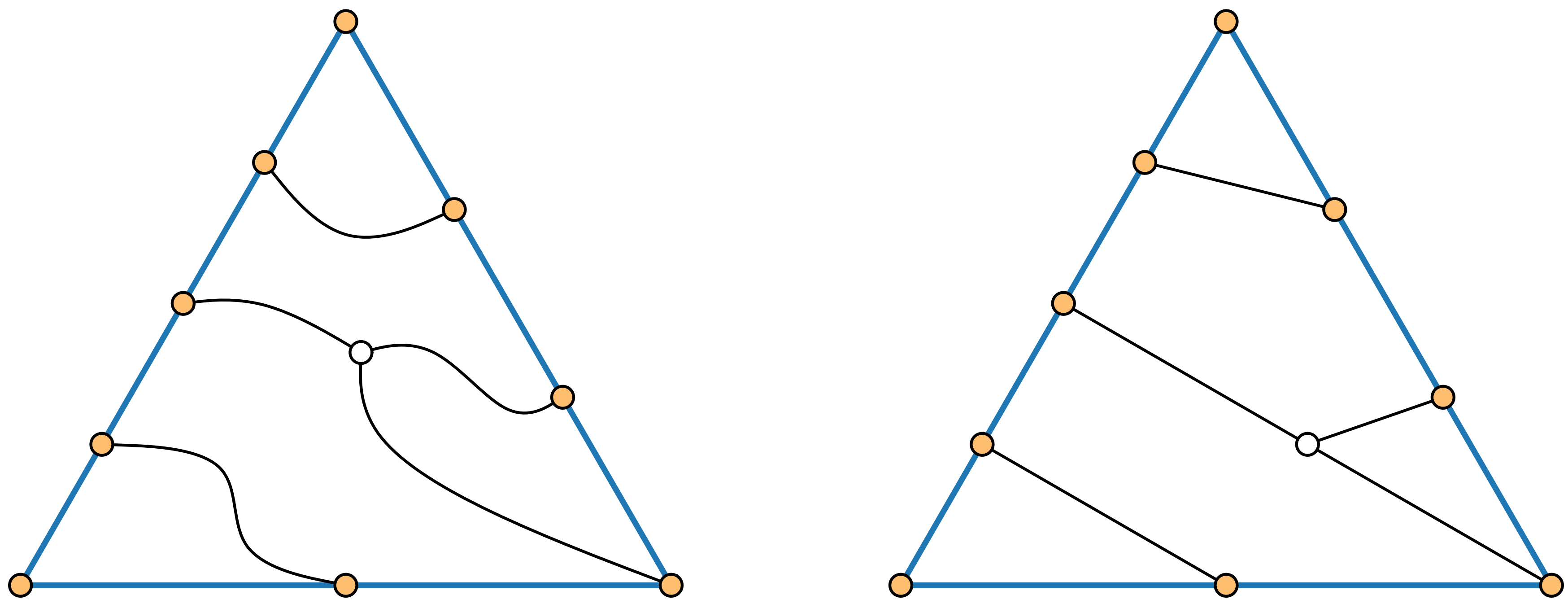


Windmill algorithm

Algorithm

Pre: cycle C drawn convex

Post: inside of C drawn

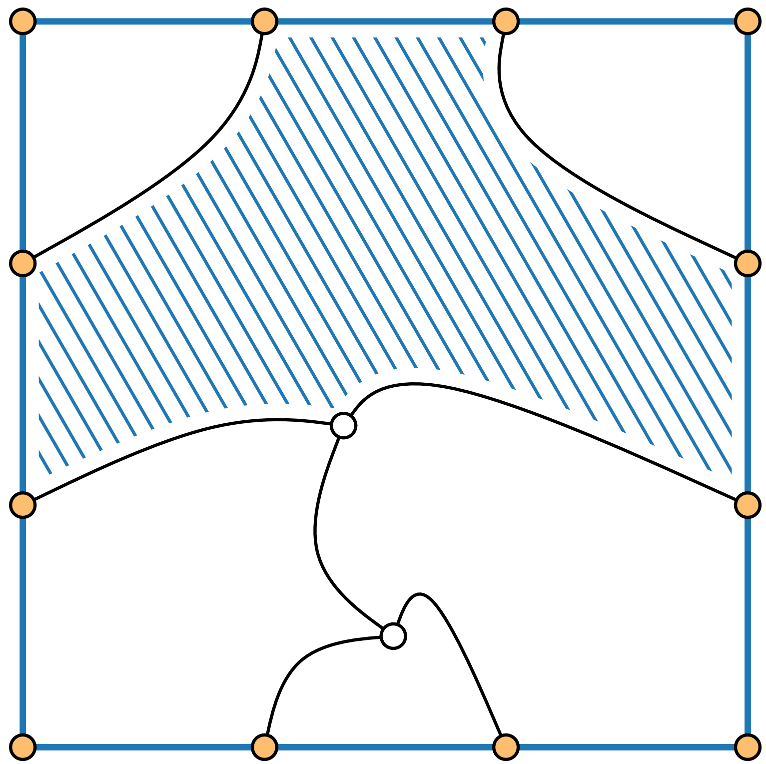


Windmill algorithm

Algorithm

Pre: cycle C drawn convex

Post: inside of C drawn

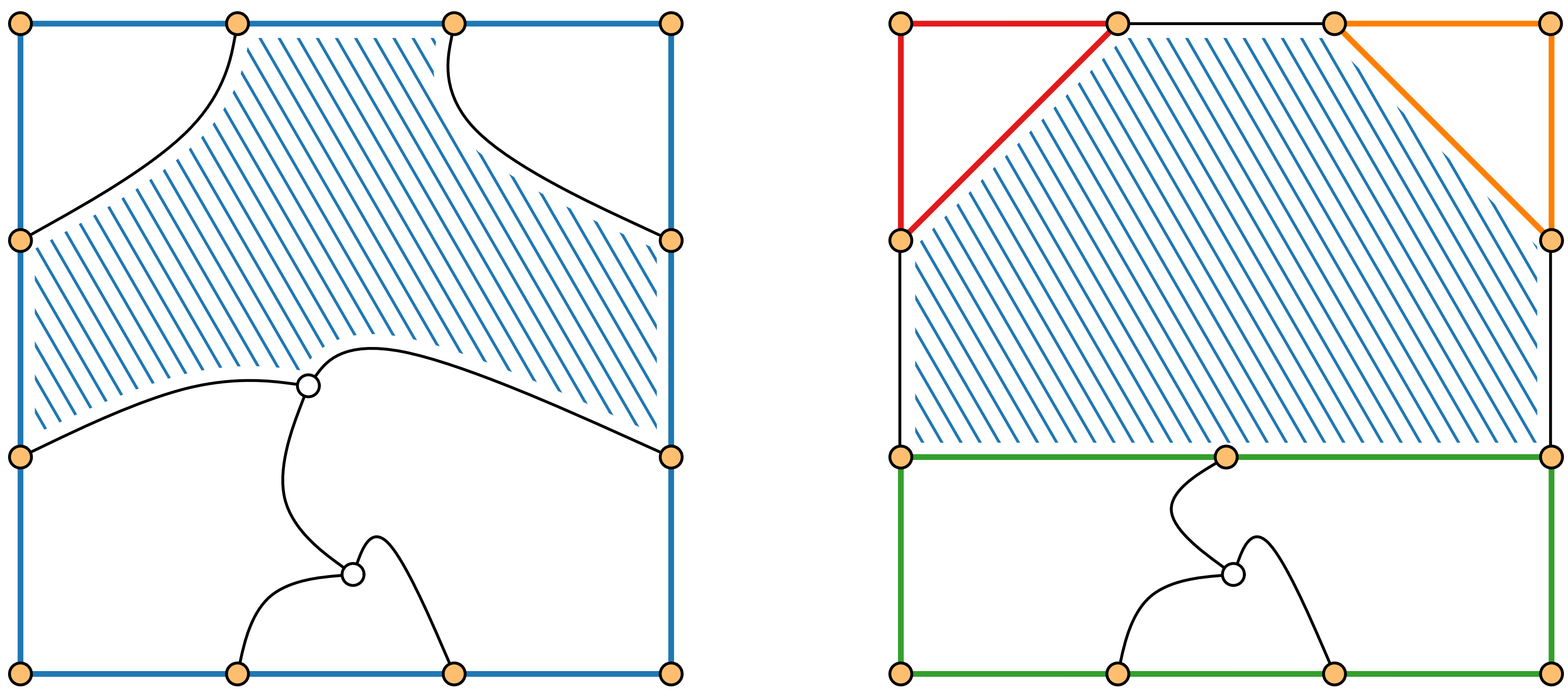


Windmill algorithm

Algorithm

Pre: cycle C drawn convex

Post: inside of C drawn

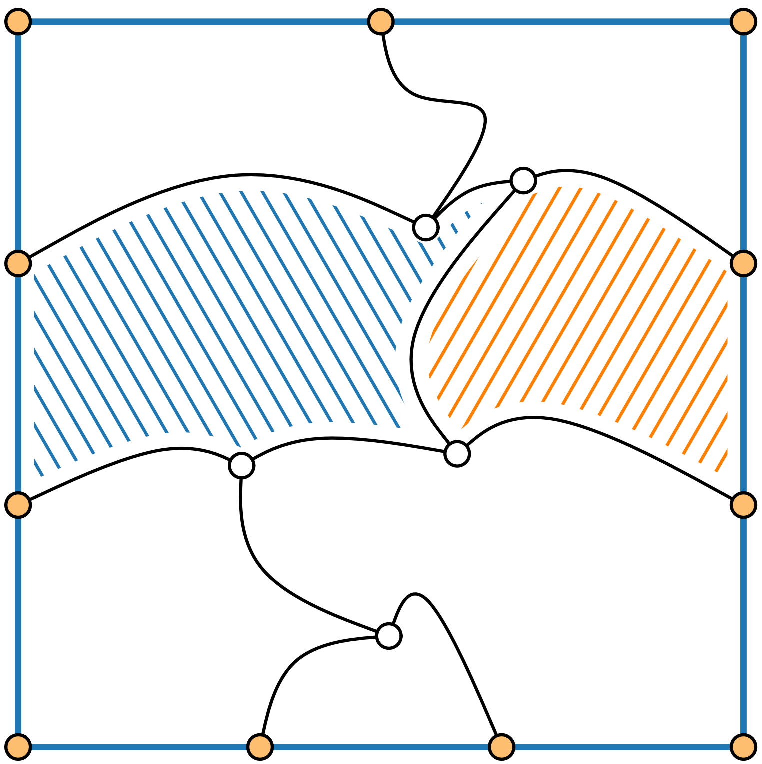


Windmill algorithm

Algorithm

Pre: cycle C drawn convex

Post: inside of C drawn

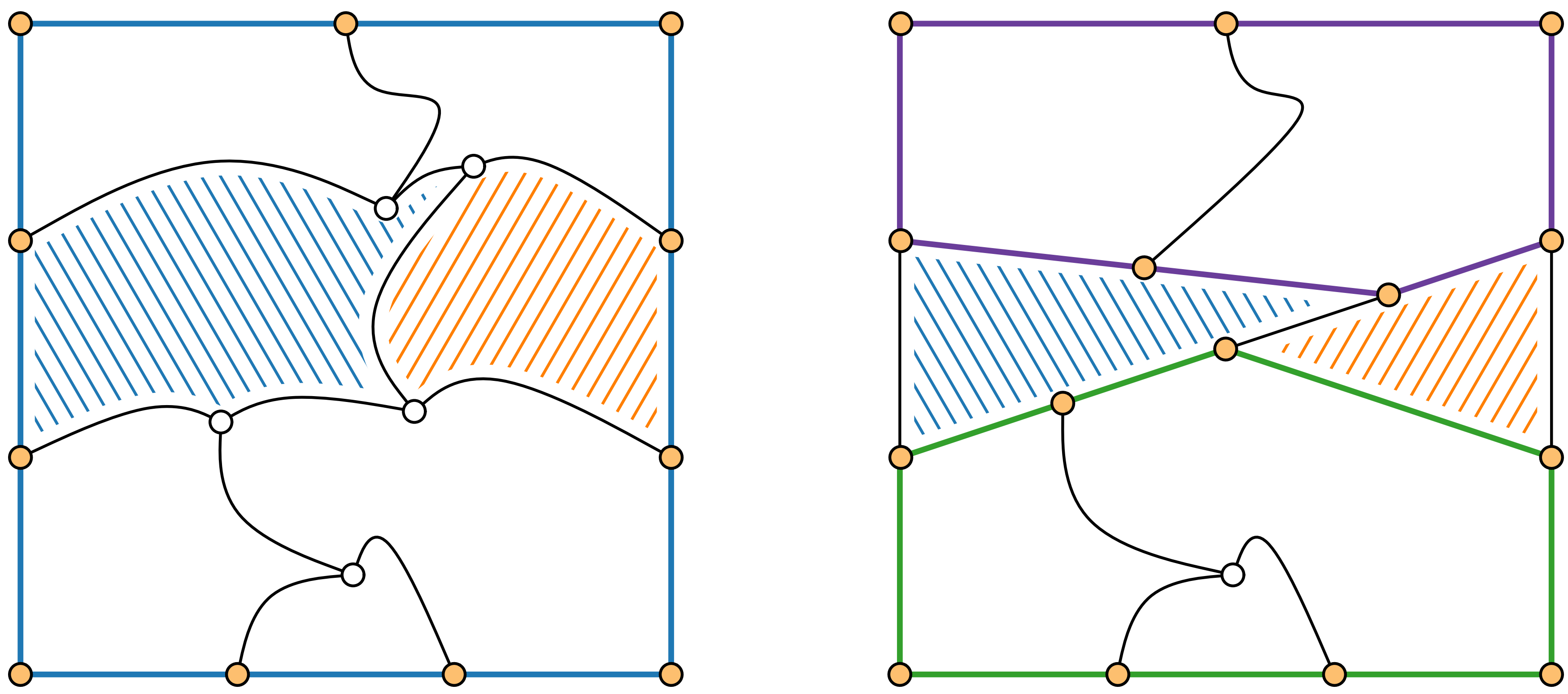


Windmill algorithm

Algorithm

Pre: cycle C drawn convex

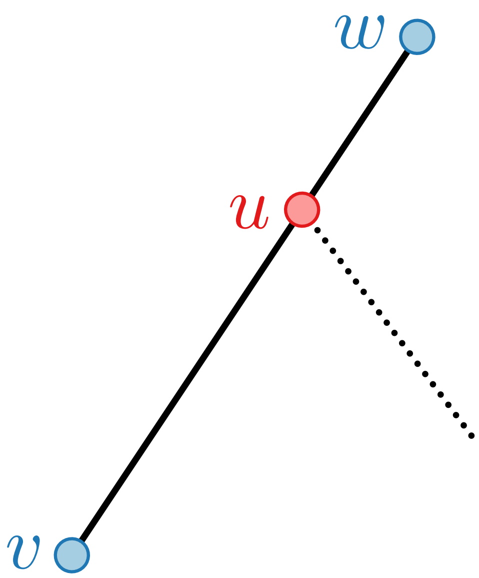
Post: inside of C drawn



Postprocessing

Set of harmonic equations [Aerts & Felsner, 2013]

$u = \lambda v + (1 - \lambda)w$, for $\lambda \in (0, 1)$

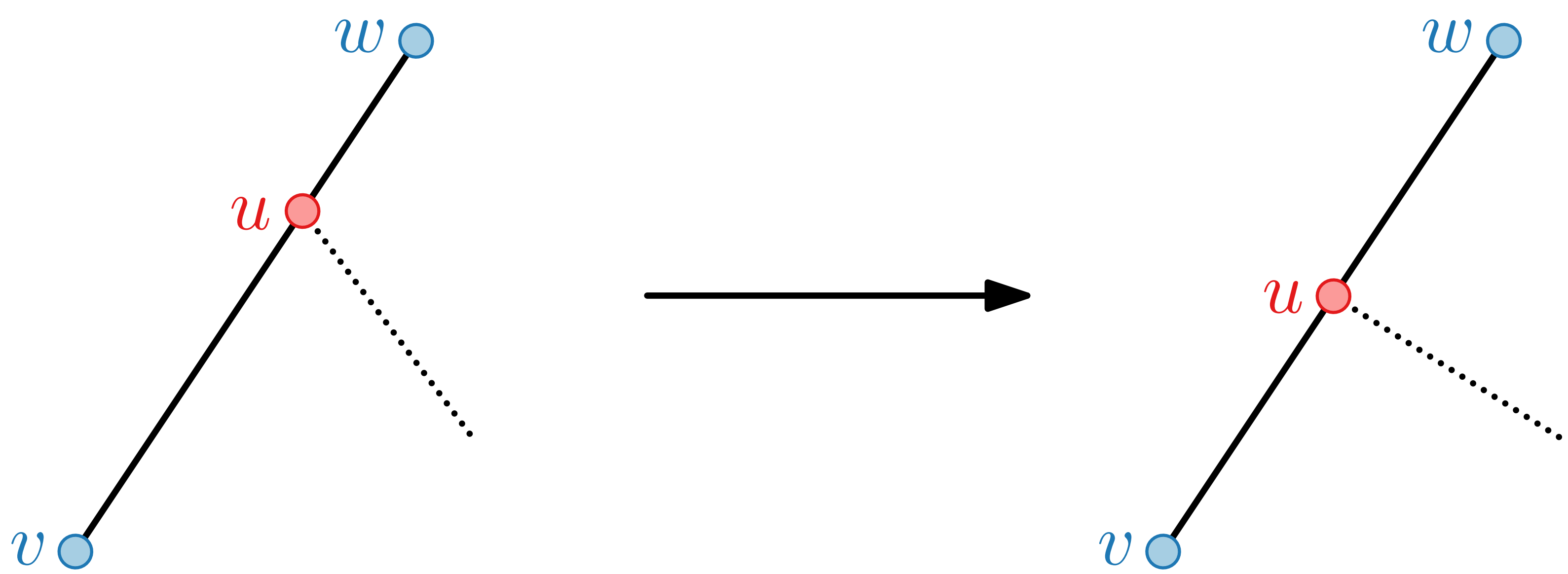


Postprocessing

Set of harmonic equations [Aerts & Felsner, 2013]

$$u = \lambda v + (1-\lambda)w, \text{ for } \lambda \in (0,1)$$

Solve for **uniform edge length**, i.e. $\lambda = 1/2$



[Mondal et al, 2013]

[Mondal et al, 2013]

"Grid"

$n/2 + 4$ segments

6 slopes

$(n/2 + 1)^2$ grid

[Mondal et al, 2013]

"Grid"

$n/2 + 4$ segments

6 slopes

$(n/2 + 1)^2$ grid

Resolved flaw in algorithm

[Mondal et al, 2013]

"Grid"

$n/2 + 4$ segments

6 slopes

$(n/2+1)^2$ grid

Resolved flaw in algorithm

"Min"

$n/2 + 3$ segments

7 slopes

Not on a grid

[Mondal et al, 2013]

“Grid”

$n/2 + 4$ segments

6 slopes

$(n/2 + 1)^2$ grid

Resolved flaw in algorithm

“Min”

$n/2 + 3$ segments

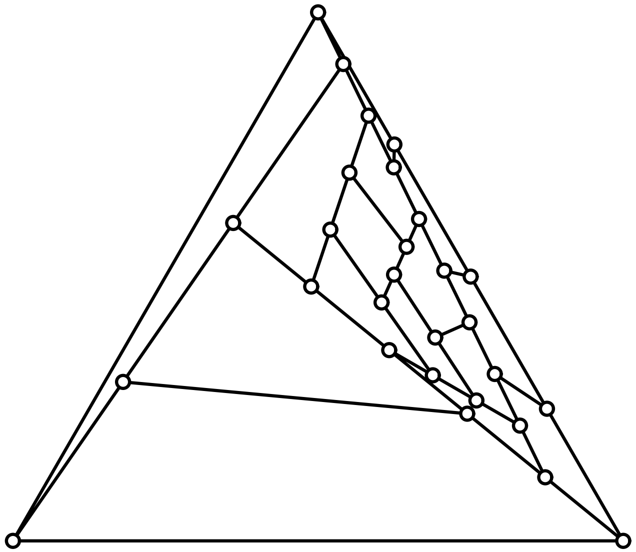
7 slopes

Not on a grid

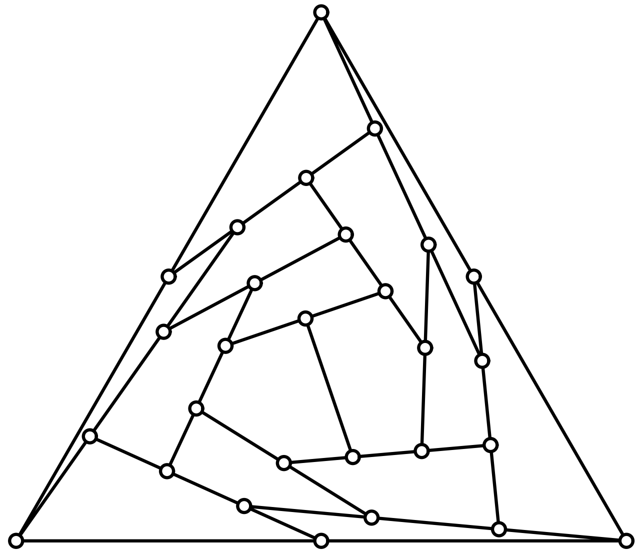
Reduced to 6 slopes

On a grid

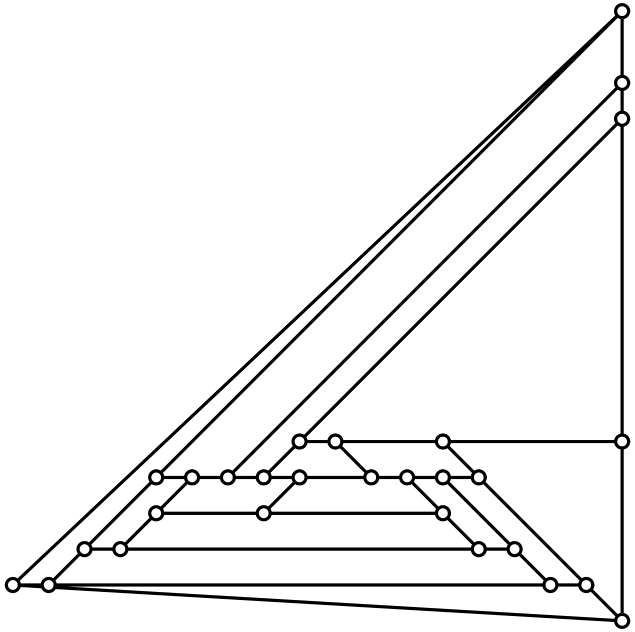
Three algorithms



Deconstruction



Windmill



[Mondal et al, 2013]

Measuring layout quality

2000 graphs with $24 \ldots 30$ vertices
using plantri

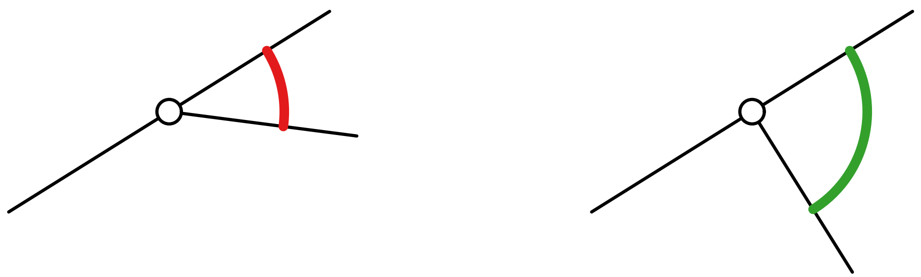
Six measures for each graph-algorithm pair

Measuring layout quality

2000 graphs with $24 \dots 30$ vertices
using plantri

Six measures for each graph-algorithm pair

Angular resolution

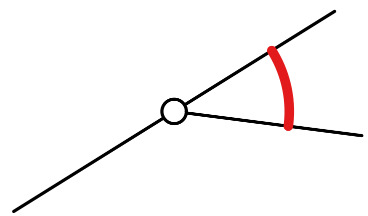
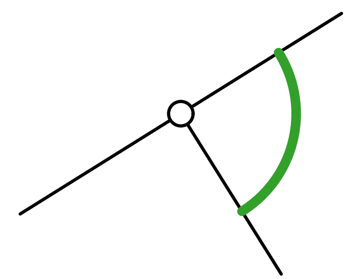


Measuring layout quality

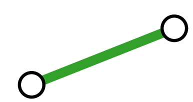
2000 graphs with $24 \ldots 30$ vertices
using plantri

Six measures for each graph-algorithm pair

Angular resolution

Edge length

Measuring layout quality

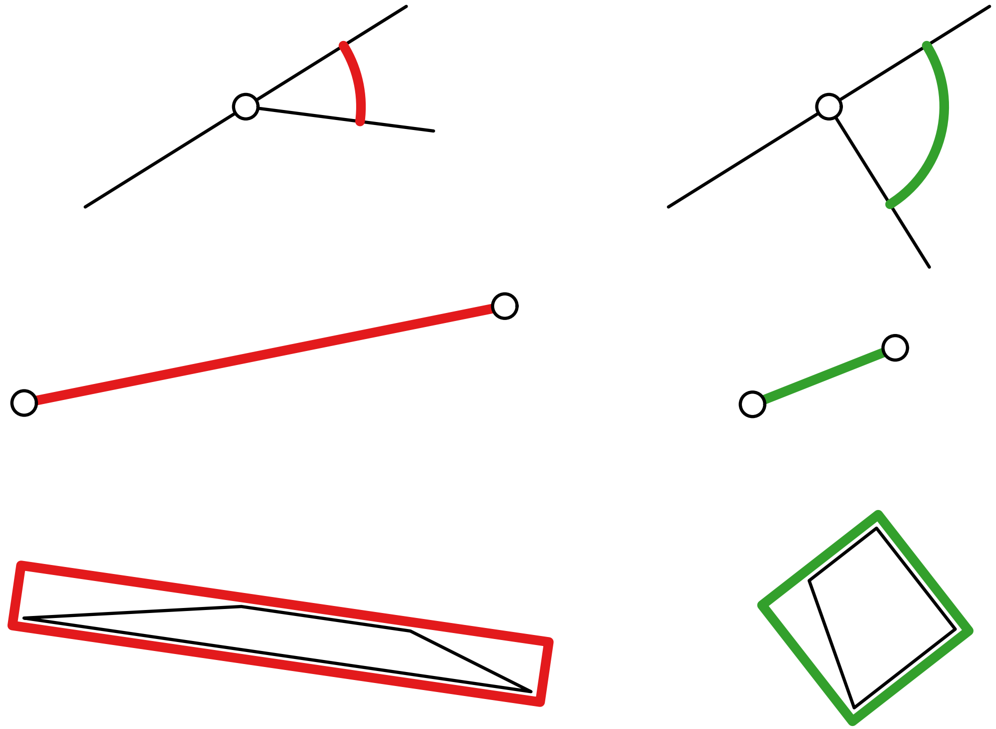
2000 graphs with $24 \ldots 30$ vertices
using plantri

Six measures for each graph-algorithm pair

Angular resolution

Edge length

Face aspect ratio

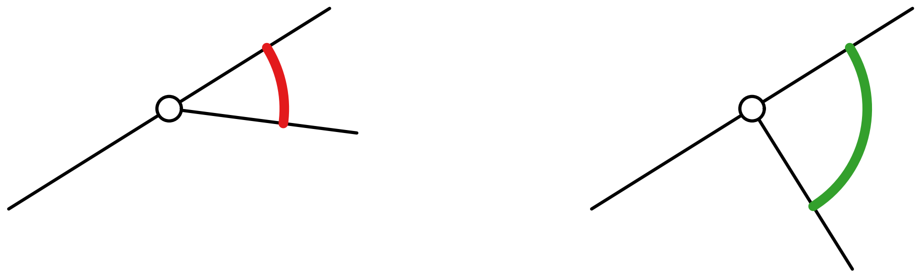


Measuring layout quality

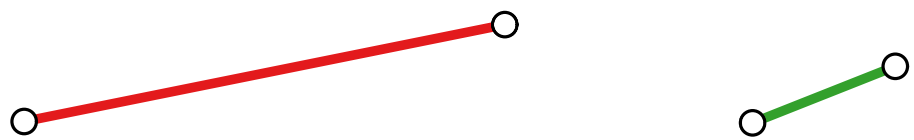
2000 graphs with $24 \ldots 30$ vertices
using plantri

Six measures for each graph-algorithm pair

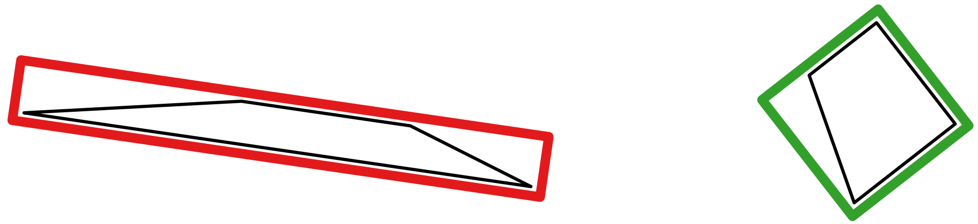
Angular resolution



Edge length

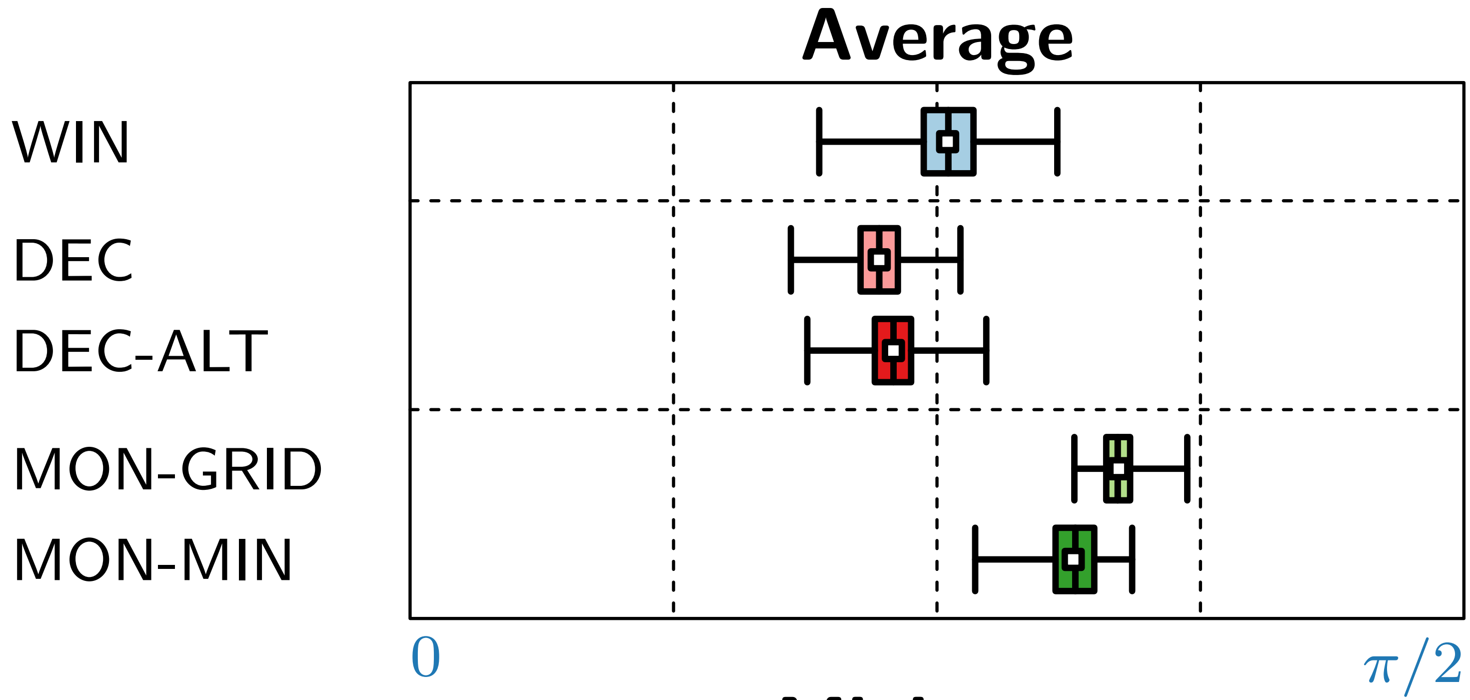


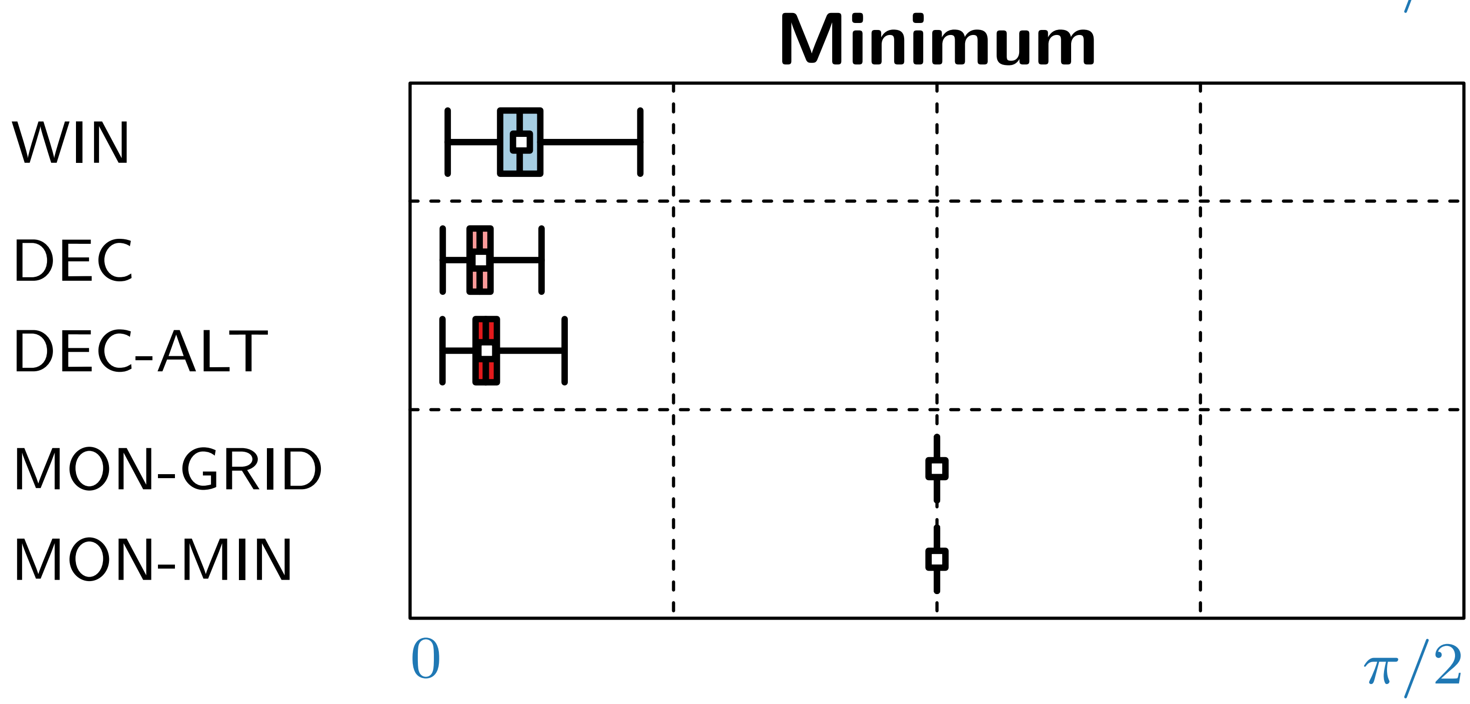
Face aspect ratio



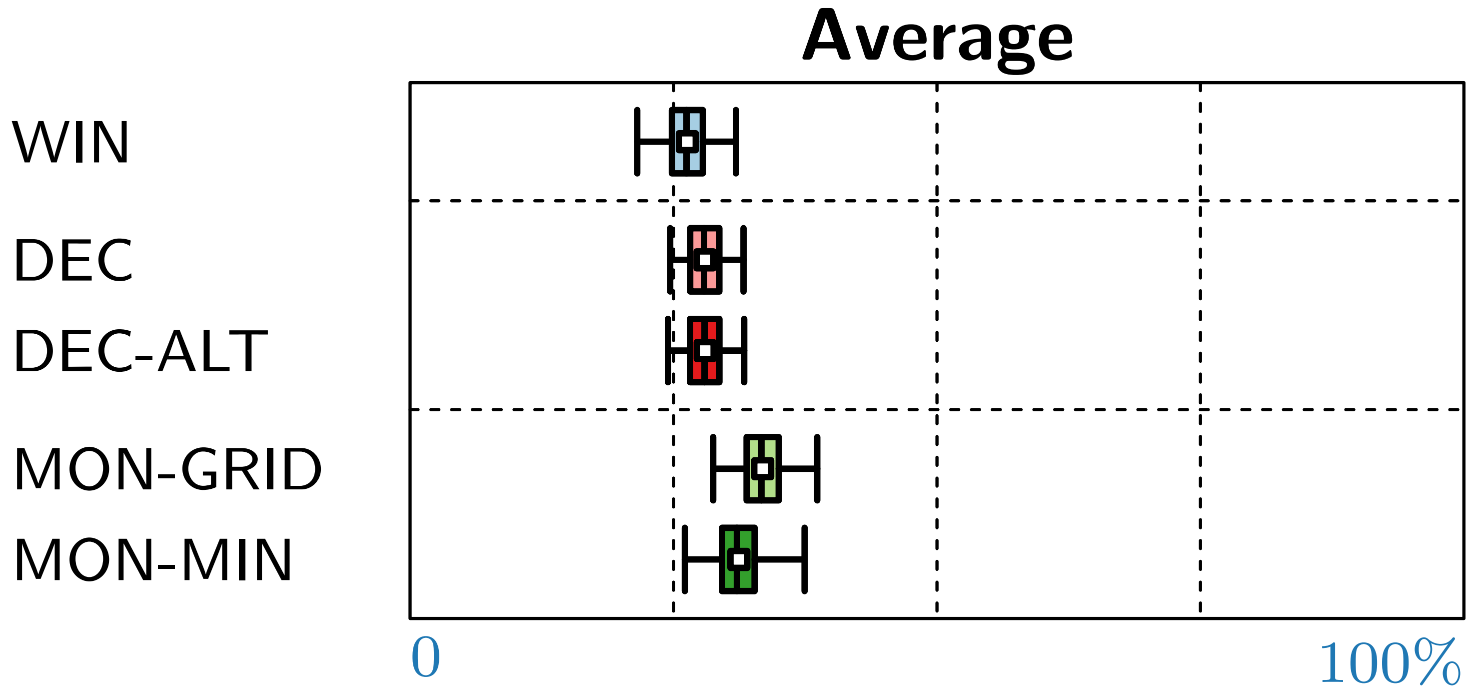
Average and worst-case

Angular resolution

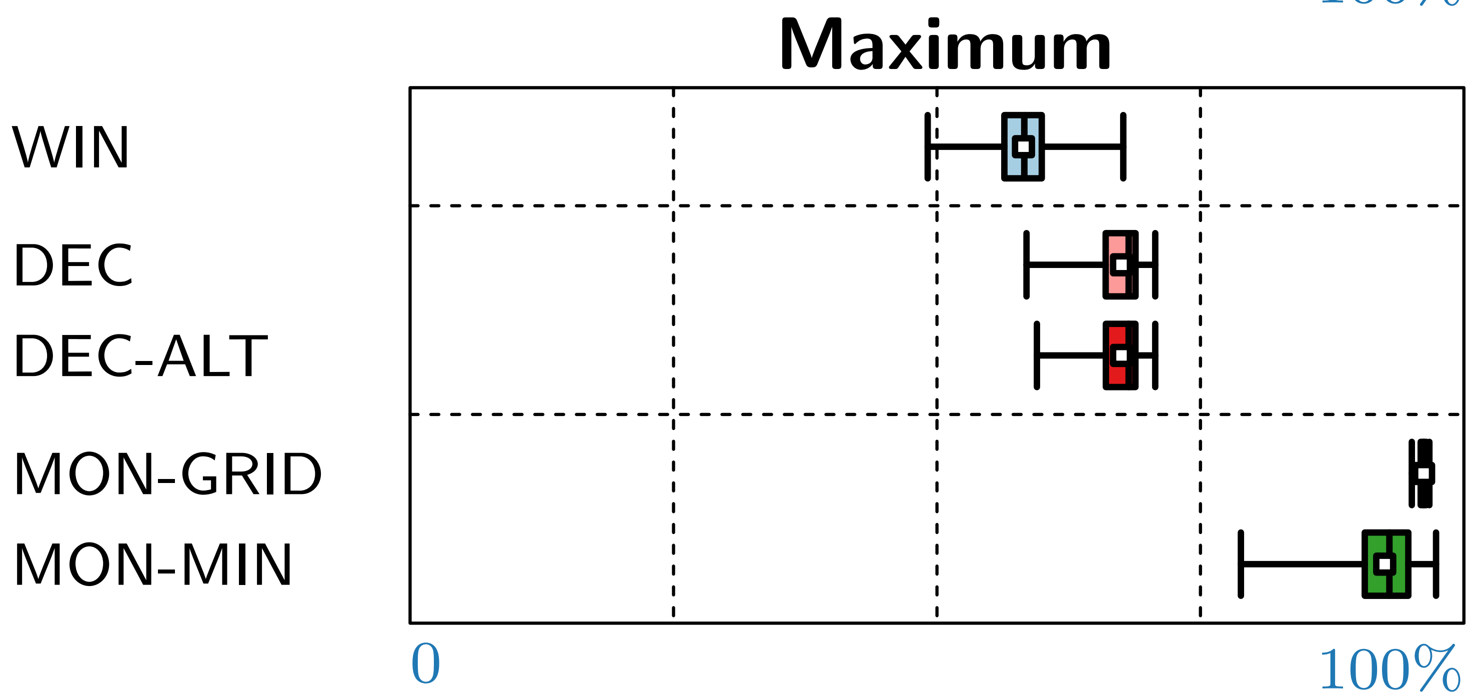




Edge length

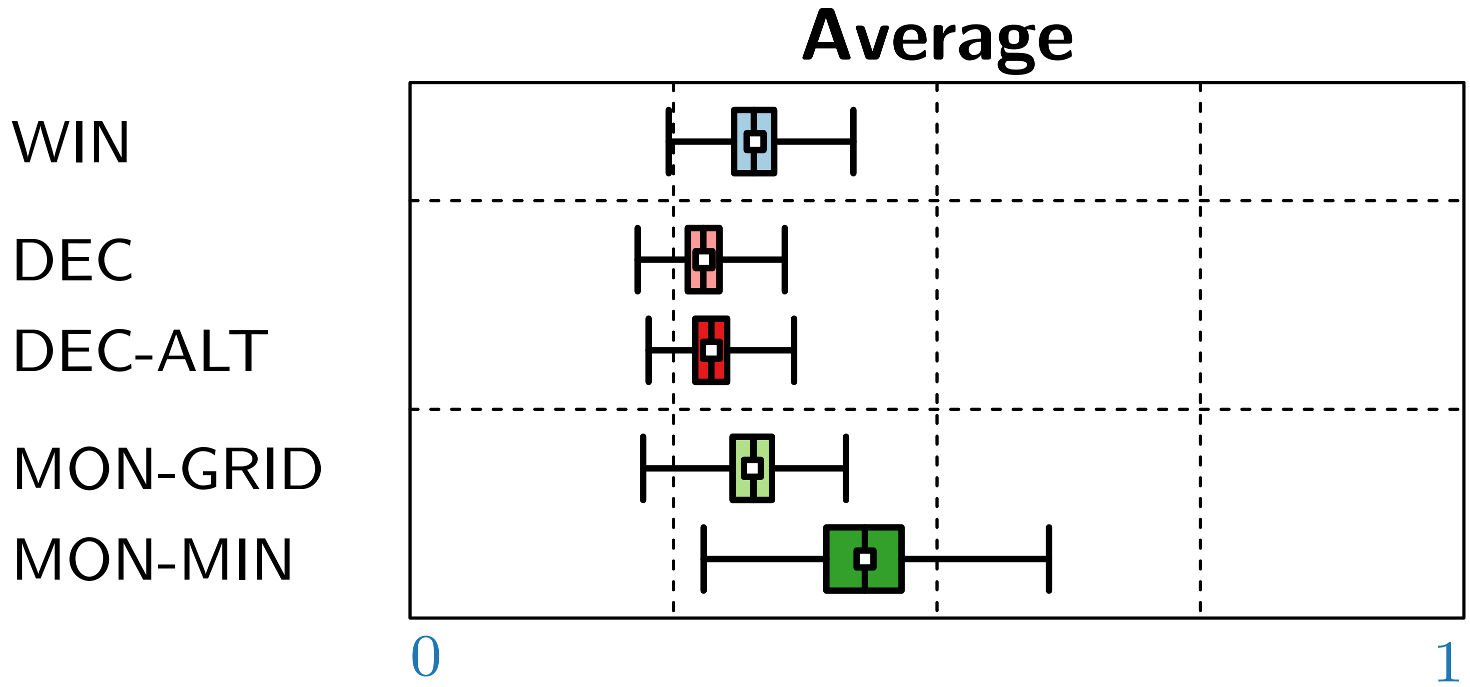


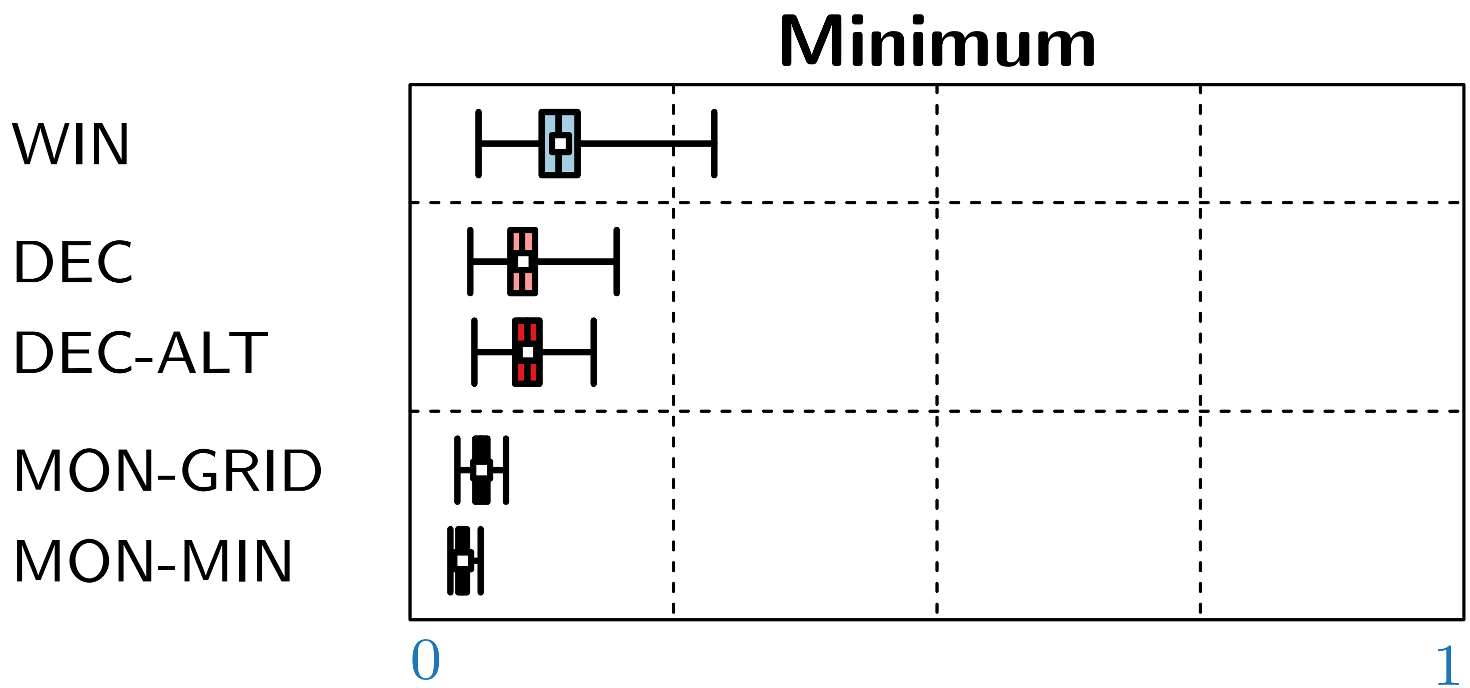
Average
WIN
DEC
DEC-ALT
MON-GRID
MON-MIN
0
100%



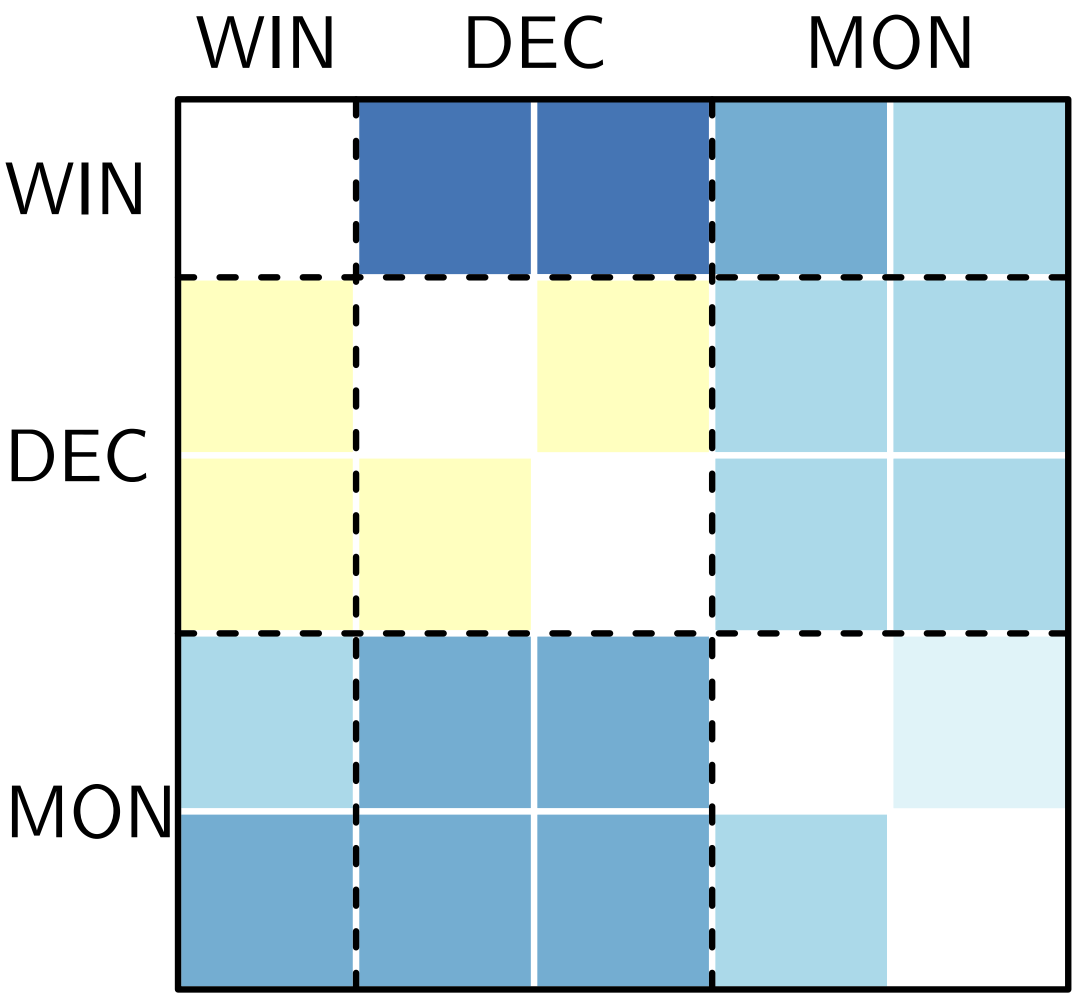
Maximum
WIN
DEC
DEC-ALT
MON-GRID
MON-MIN
0
100%

Face aspect ratio

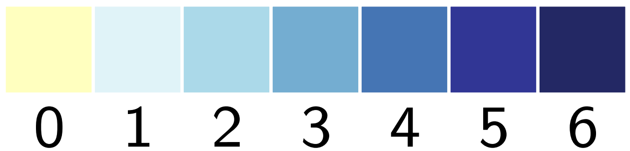




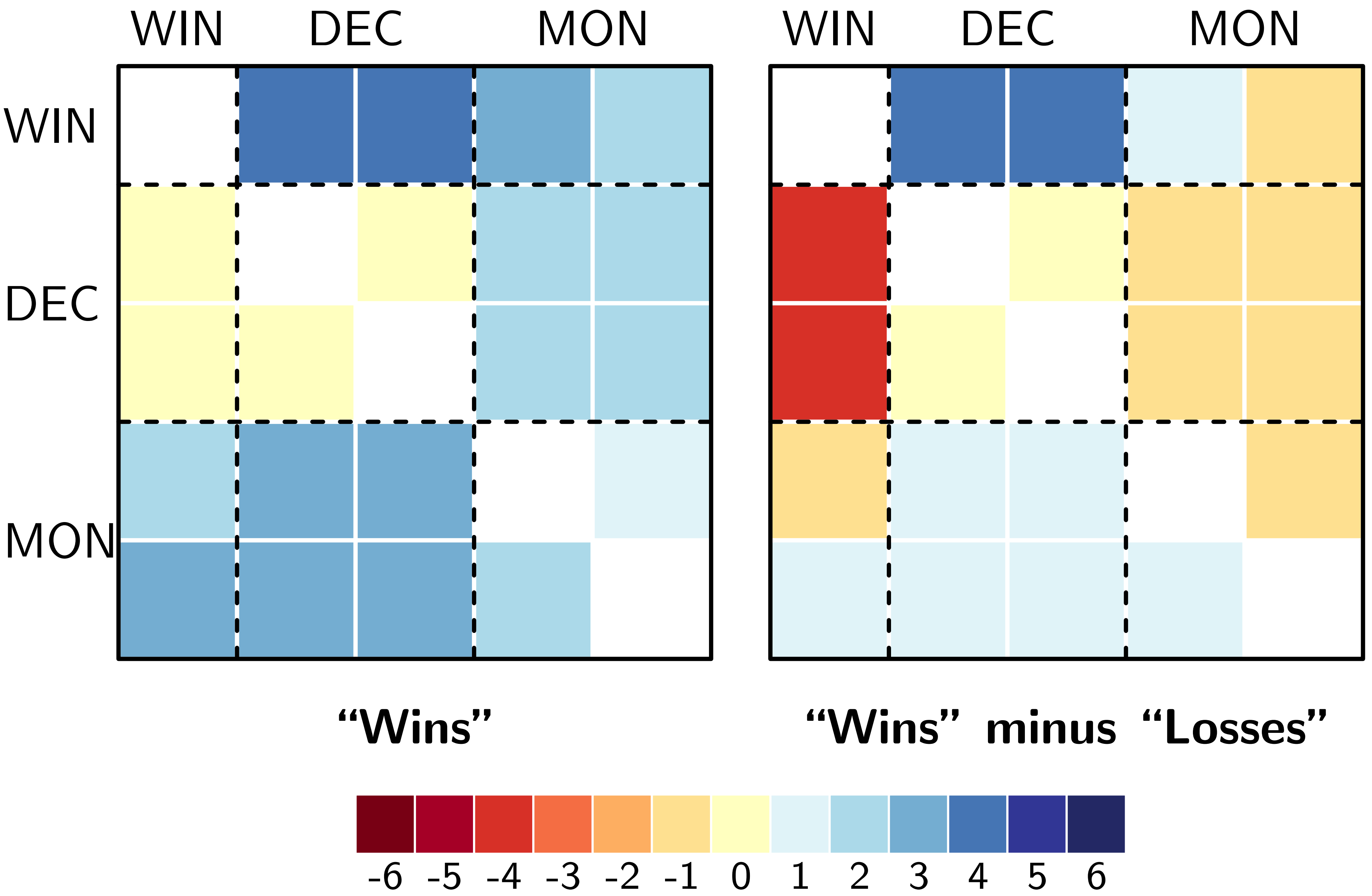
Experiment summary



“Wins”



Experiment summary

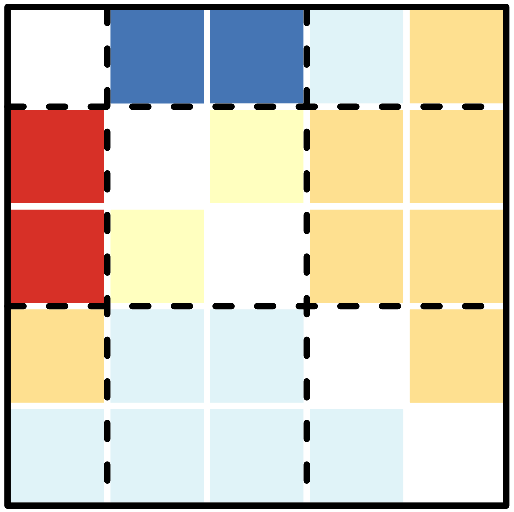


Conclusion

Minimal visual complexity

Two new algorithms

Fixed and improved [Mondal et al, 2013]



Experiments

Best depends on measure

Conclusion

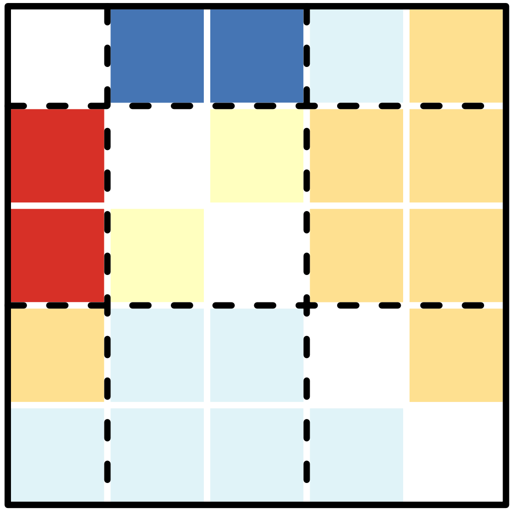
Minimal visual complexity

Two new algorithms

Fixed and improved [Mondal et al, 2013]

Experiments

Best depends on measure



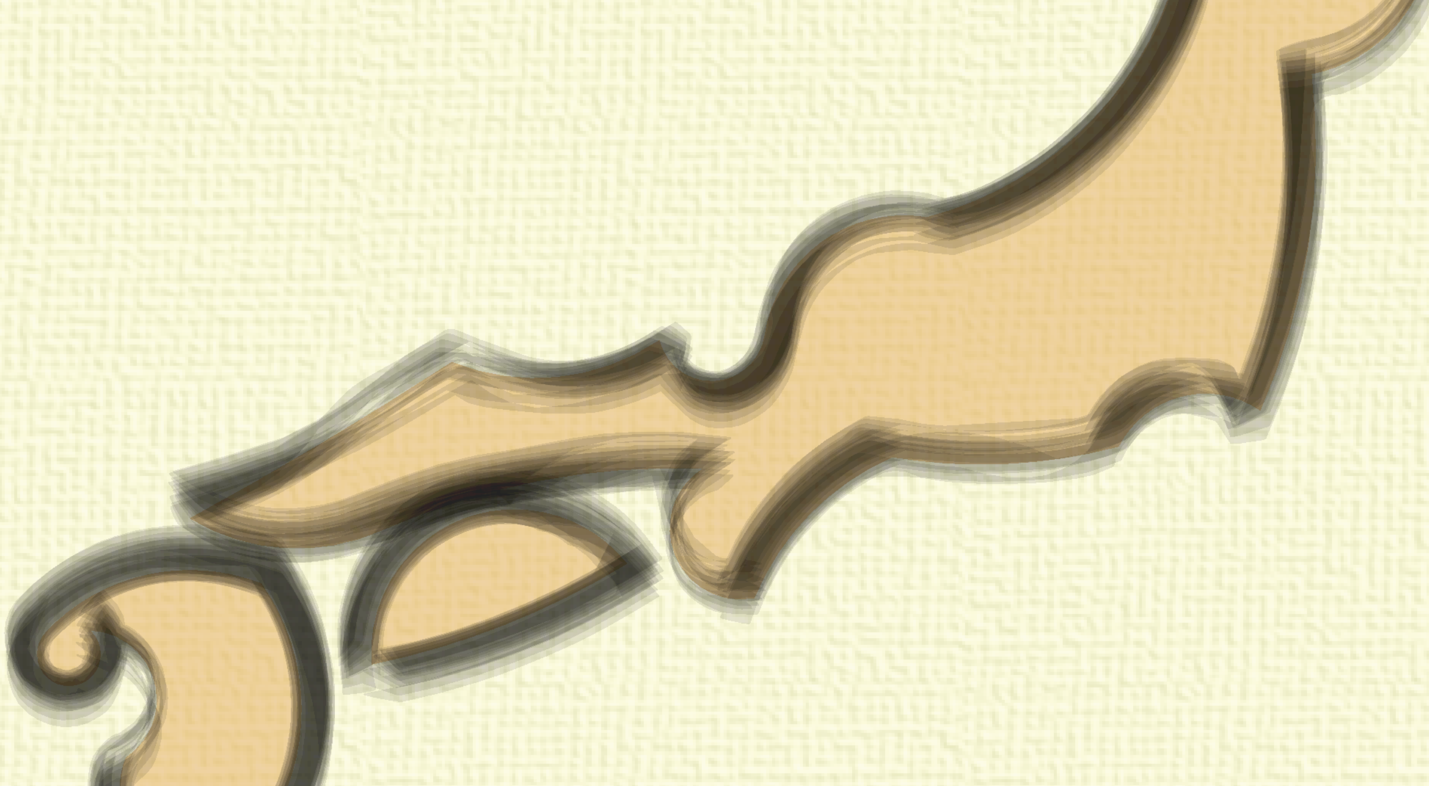
Future work

Closing gap for other classes

Circular arcs

Visual complexity $\sim$ observer's assessment?

Visual complexity $\sim$ cognitive load?



Thank you for listening!

Wouter Meulemans <*wouter.meulemans@city.ac.uk*>

[Van Goethem, Meulemans, Speckmann, Wood, *TVCG*, 2015]

[Igamberdiev, Meulemans, Schulz, *GD*, 2015]

[Buchin, Meulemans, Van Renssen, Speckmann, *ACM TSAS*, to appear]