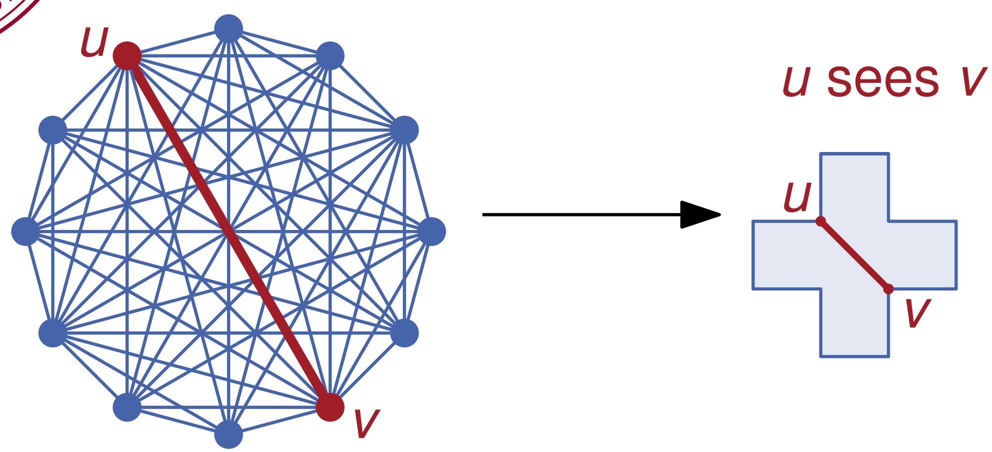
Reconstructing Generalized Staircase Polygons with Uniform Step Length

Nodari Sitchinava and **Darren Strash**

GD 2017 | Sept. 25, 2017

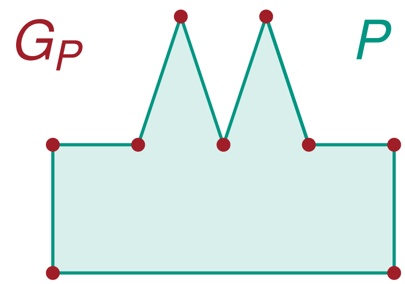
Department of Computer Science
Colgate University



Visibility graphs

Given a polygon P, we construct its *visibility graph* G_P as follows:

- vertex v in $G_P \leftrightarrow v$ is a vertex on P's boundary.
- edge $(u, v) \leftrightarrow u$ sees v. (uv does not intersect exterior of P)

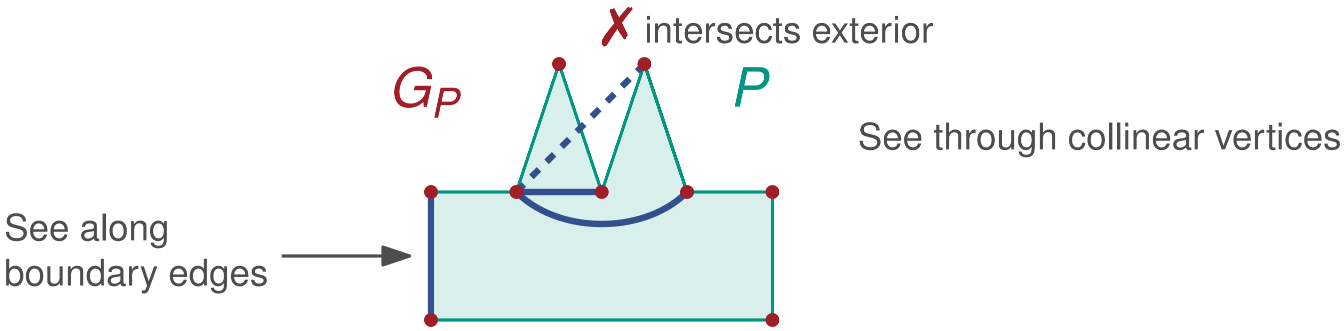


Nodari Sitchinava and Darren Strash:
Reconstructing Generalized Staircase Polygons with Uniform Step Length
Department of Computer Science
Colgate University

Visibility graphs

Given a polygon P, we construct its *visibility graph* G_P as follows:

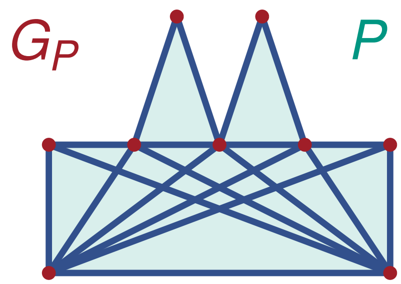
- vertex v in $G_P \leftrightarrow v$ is a vertex on P's boundary.
- edge $(u, v) \leftrightarrow u$ sees v. (uv does not intersect exterior of P)



Visibility graphs

Given a polygon P, we construct its *visibility graph* G_P as follows:

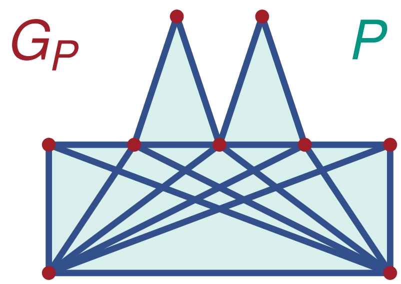
- vertex v in $G_P \leftrightarrow v$ is a vertex on P's boundary.
- edge $(u, v) \leftrightarrow u$ sees v. (uv does not intersect exterior of P)



Visibility graphs

Given a polygon P, we construct its *visibility graph* G_P as follows:

- vertex v in $G_P \leftrightarrow v$ is a vertex on P's boundary.
- edge $(u, v) \leftrightarrow u$ sees v. (uv does not intersect exterior of P)

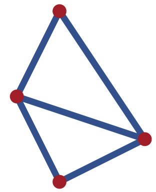


Can be computed in $O(n \log n + m)$ time [Ghosh & Mount, 1991]

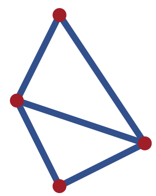
What about the reverse?

Recognition and reconstruction

Input: A graph G:

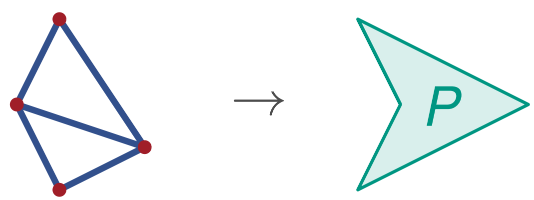


- **Recognition Problem:** Is G the visibility graph of *some* polygon?



$\rightarrow \quad \exists P?$

- **Reconstruction Problem:** Give a polygon P, which has G has its visibility graph.

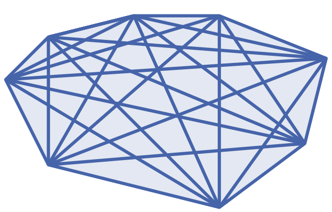


Quick: Known Results

- Recognition is in PSPACE. [Everett, 1994]
- Reconstruction complexity is open
- Special cases: limited visibility due to reflex chains

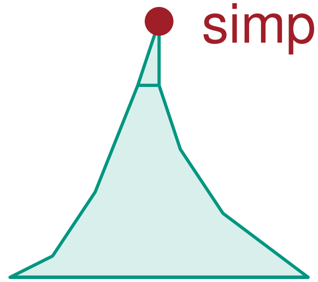
A single clique

convex



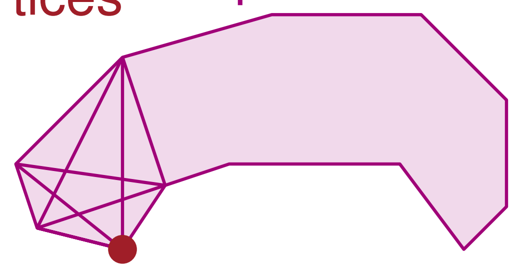
tower

simplex vertices



[Choi et al., 1995]

spiral

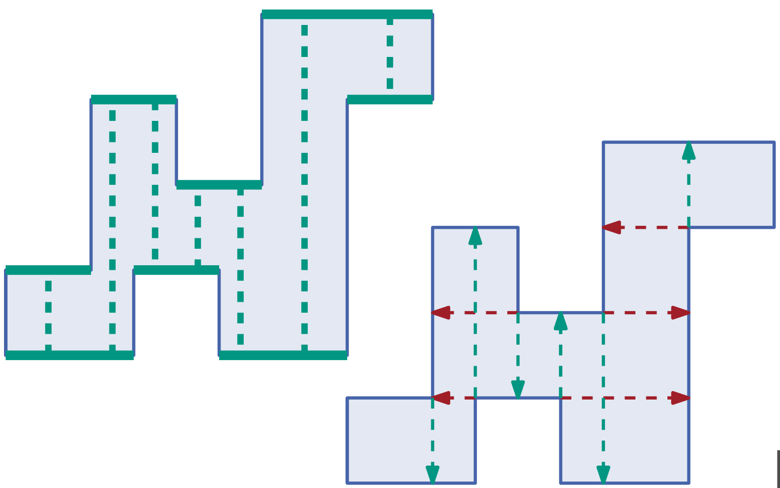


[Everett & Corneil, 1995]

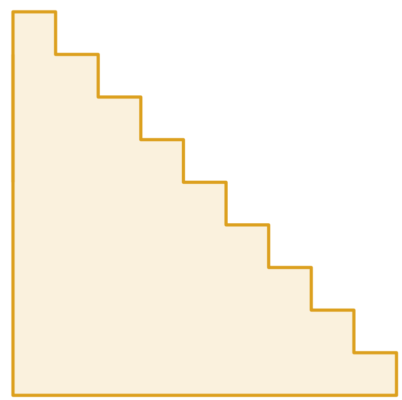
Quick: Known Results

- Recognition is in PSPACE. [Everett, 1994]
- Reconstruction complexity is open
- Special cases: limited visibility due to reflex chains
- Orthogonal polygons?

Alternative visibility definitions

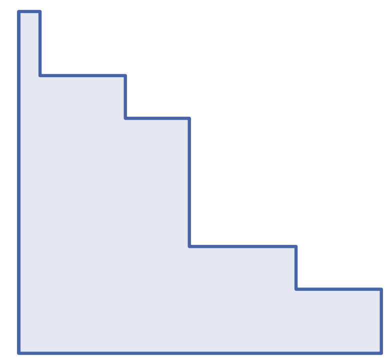


Reconstruction



[Abello & Eğecioğlu, 1993]

Recognition

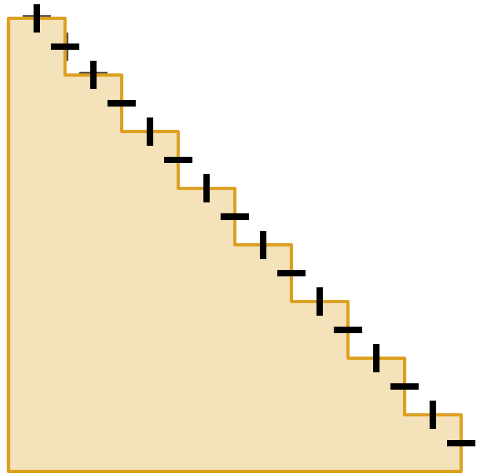


[Abello et al., 1995]

Uniform-Step Length Polygons

The *only* reconstruction result is for staircase polygons with uniform length [Abello & Eğecioğlu, 1993]

All "steps" have same length

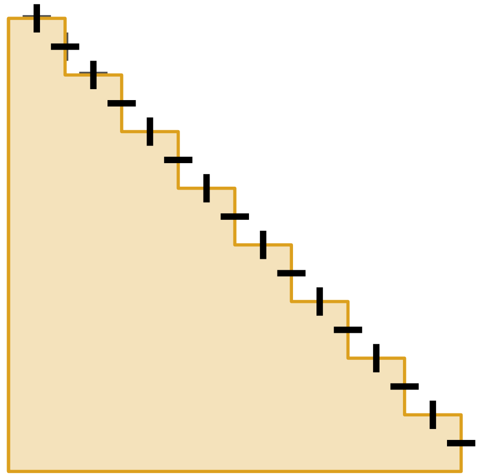


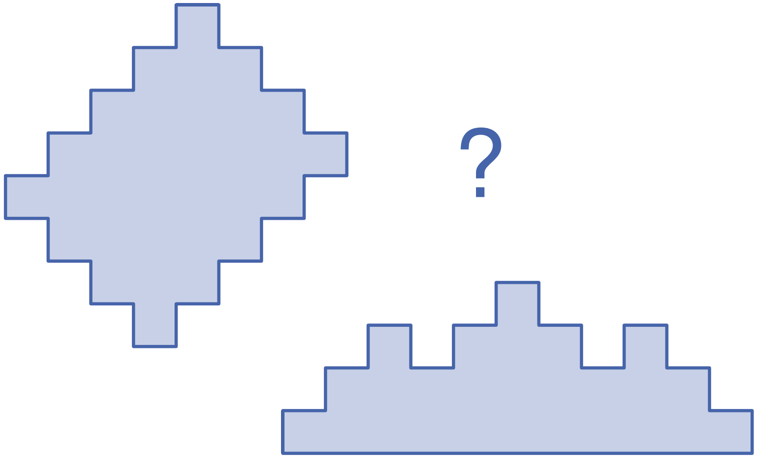
Uniform-Step Length Polygons

The *only* reconstruction result is for staircase polygons with uniform length [Abello & Eğecioğlu, 1993]

Can we *efficiently* reconstruct polygons with more staircases?

All “steps” have same length

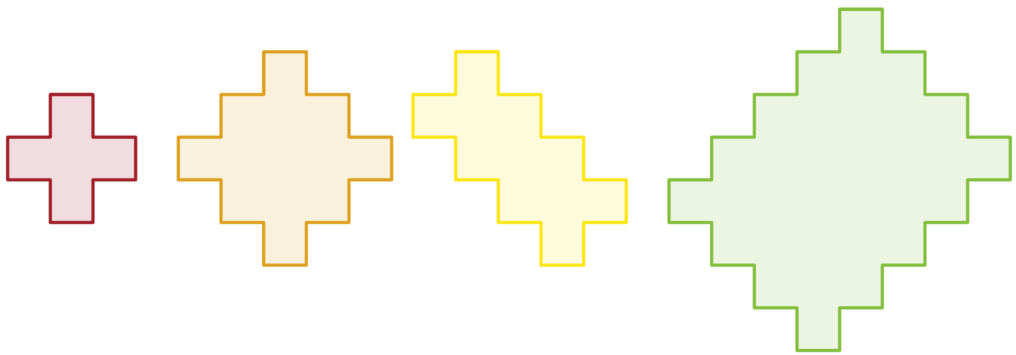




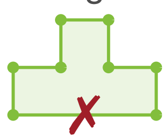
Our Results

Uniform-length orthogonally convex polygons can be reconstructed $O(n^2m)$ time. (n vertices, m edges.)

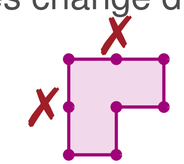
unit-length orthogonally convex



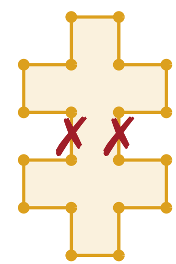
same edge lengths



edges change direction



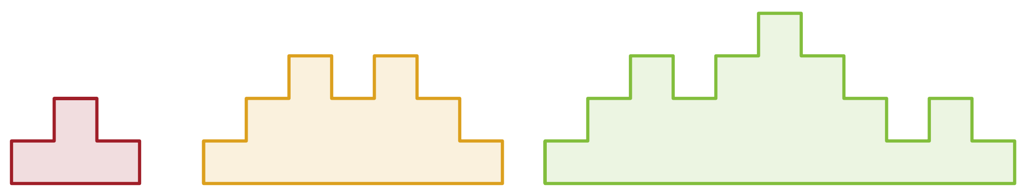
no dents

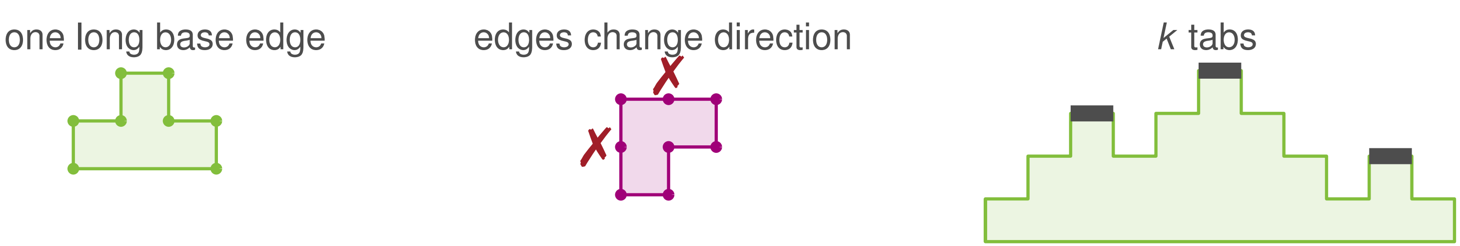


Our Results

Histogram reconstruction is fixed parameter tractable on the number of tabs k, with time $O(n^2 m + (k-2)!2^{k-2}(n \log n + m))$.

uniform-length histogram

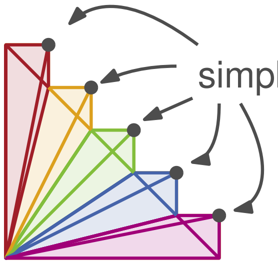




Simplicial vertices and edges

Maximal clique in $G_P \leftrightarrow$ **maximal** convex subpolygon on vertices of P.

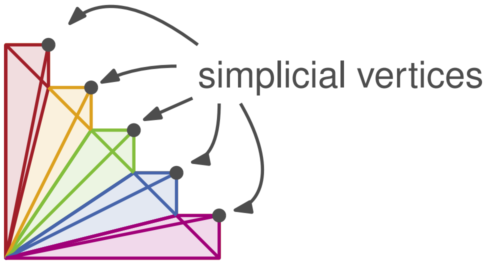
- Staircase polygons have simplicial vertices



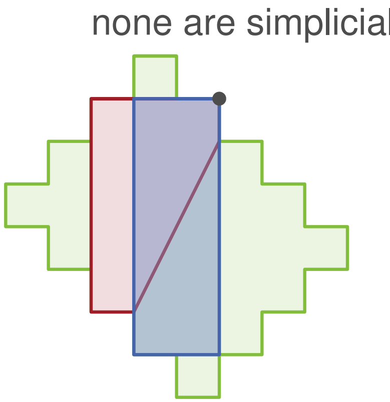
Simplicial vertices and edges

Maximal clique in $G_P \leftrightarrow$ **maximal** convex subpolygon on vertices of P.

- Staircase polygons have simplicial vertices



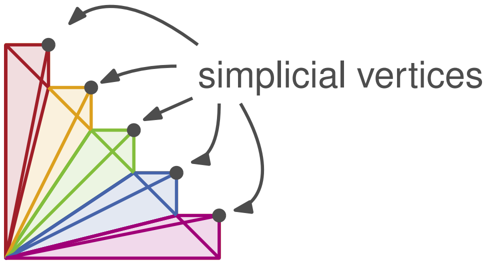
- No *simplicial vertices*? $\rightarrow$ we need new techniques



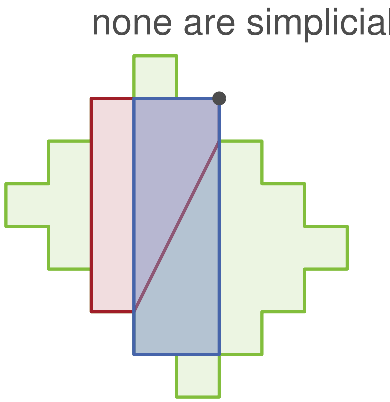
Simplicial vertices and edges

Maximal clique in $G_P \leftrightarrow$ **maximal** convex subpolygon on vertices of P.

- Staircase polygons have simplicial vertices



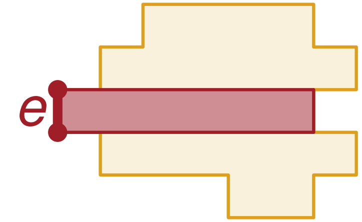
- No *simplicial vertices*? $\rightarrow$ we need new techniques



- Idea: Evaluate *edges* in only one maximal clique

  $\rightarrow$ 1-simplicial edges

  $\rightarrow$ identify convex-convex boundary edges



1-simplicial edges

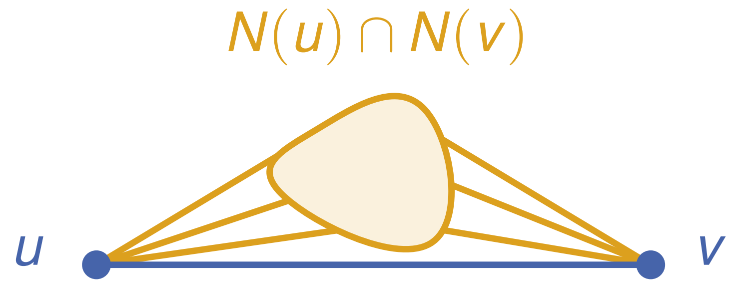
Most expensive operation is testing each edge for membership in exactly 1 maximal clique.

1-simplicial edges

Most expensive operation is testing each edge for membership in exactly 1 maximal clique.

2 Steps:

$\rightarrow$ for (u, v), find common neighborhood $O(n)$ time

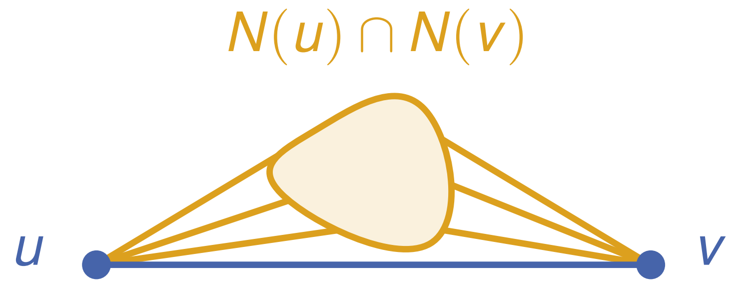


1-simplicial edges

Most expensive operation is testing each edge for membership in exactly 1 maximal clique.

2 Steps:

$\rightarrow$ for (u, v), find common neighborhood $O(n)$ time

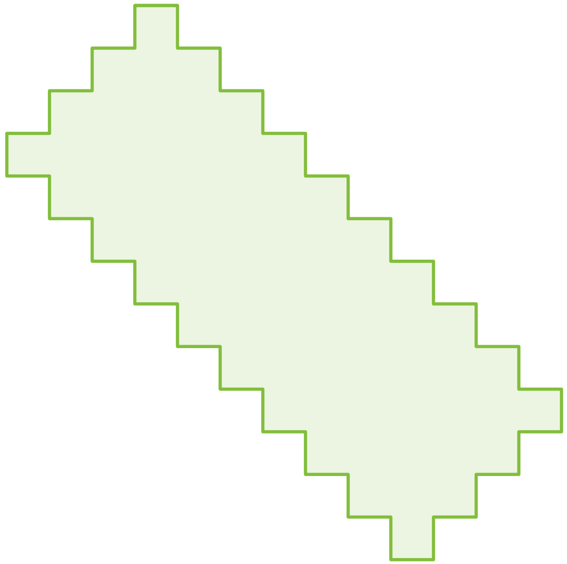


$\rightarrow$ Test if $N(u) \cap N(v)$ is a clique in $O(n^2)$ time

For all m edges, this takes $O(n^2 m)$ total time.

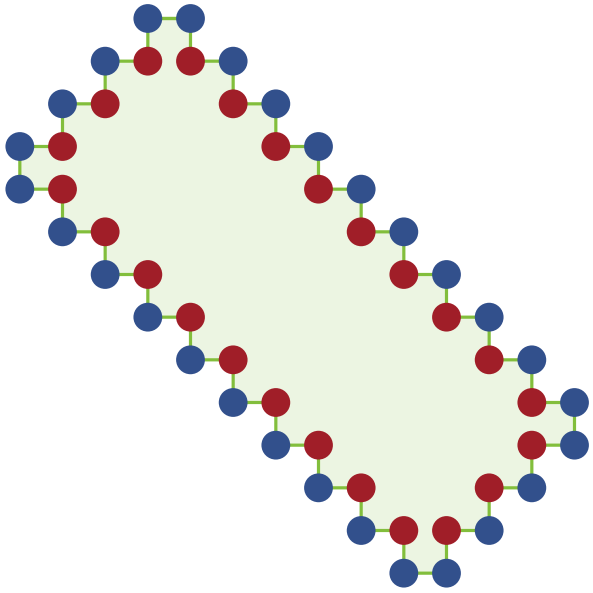
Convex case: the algorithm

Step 0: Determine convex and reflex vertices.



Convex case: the algorithm

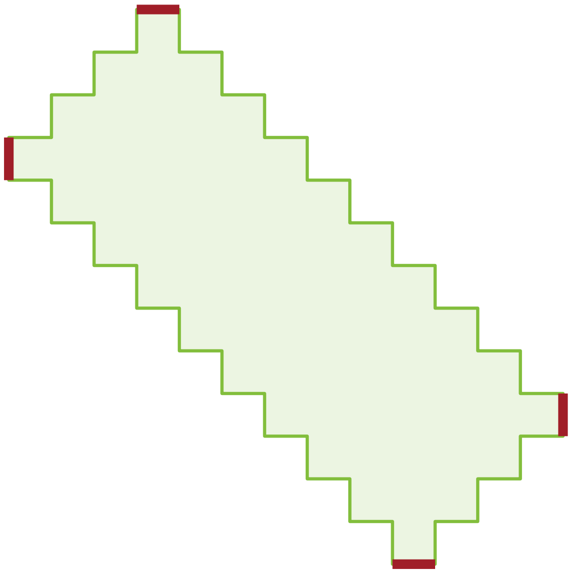
Step 0: Determine convex and reflex vertices.



Convex case: the algorithm

Step 0: Determine convex and reflex vertices.

Step 1: Find tab edges

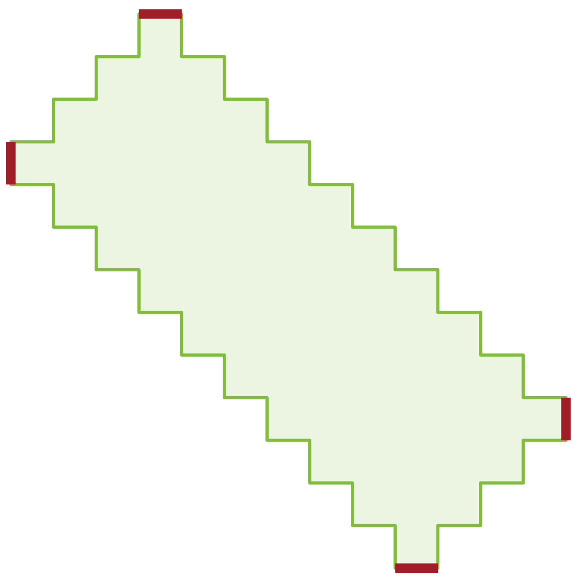


Convex case: the algorithm

Step 0: Determine convex and reflex vertices.

Step 1: Find tab edges

Step 2: Choose one to be the topmost



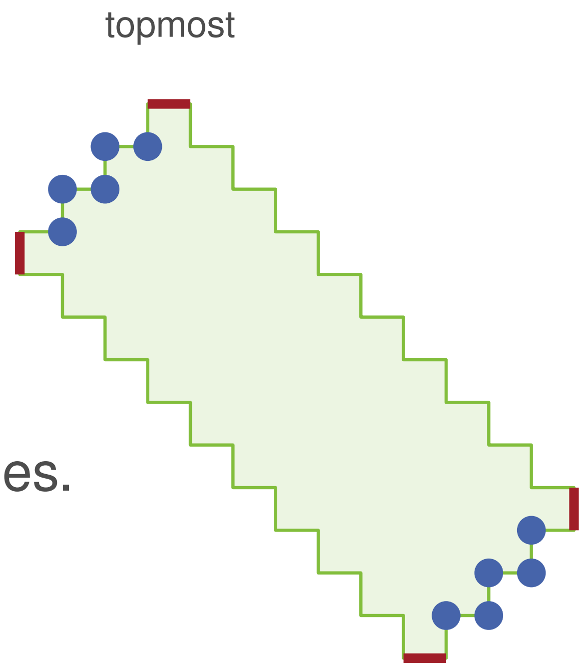
Convex case: the algorithm

Step 0: Determine convex and reflex vertices.

Step 1: Find tab edges

Step 2: Choose one to be the topmost

Step 3: Determine vertices on “short” staircases.



Convex case: the algorithm

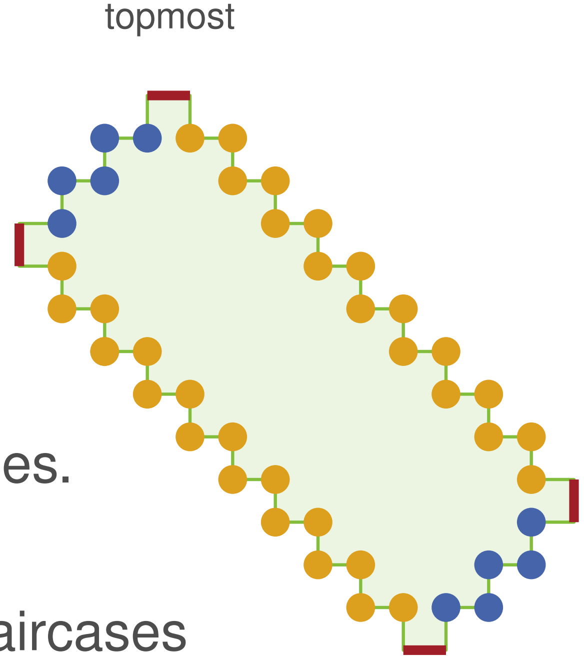
Step 0: Determine convex and reflex vertices.

Step 1: Find tab edges

Step 2: Choose one to be the topmost

Step 3: Determine vertices on “short” staircases.

Step 4: Assign remaining vertices to “long” staircases



Convex case: the algorithm

Step 0: Determine convex and reflex vertices.

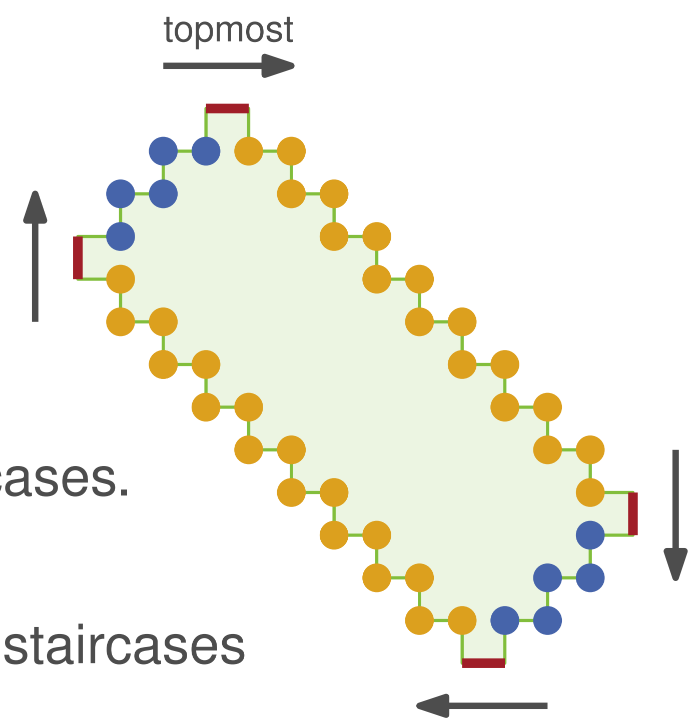
Step 1: Find tab edges

Step 2: Choose one to be the topmost

Step 3: Determine vertices on “short” staircases.

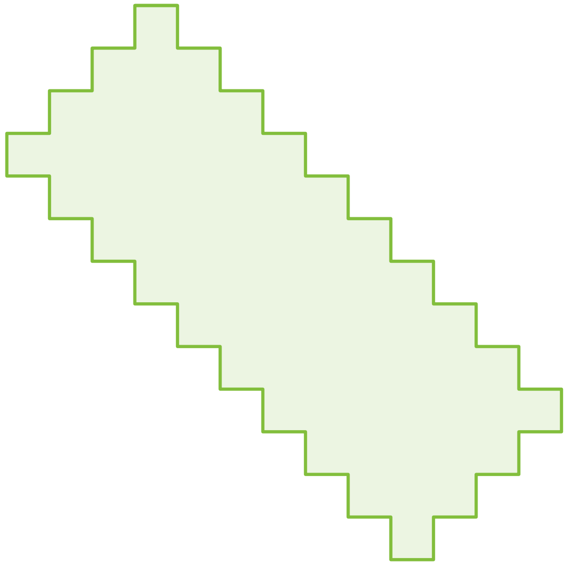
Step 4: Assign remaining vertices to “long” staircases

Step 5: Orient & construct the boundary



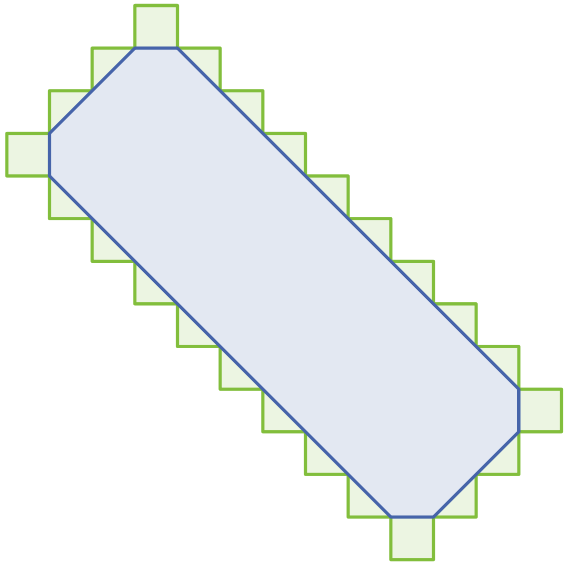
Convex/Reflex Vertices

Maximal clique in $G_P \leftrightarrow$ **maximal** convex subpolygon on vertices of P.



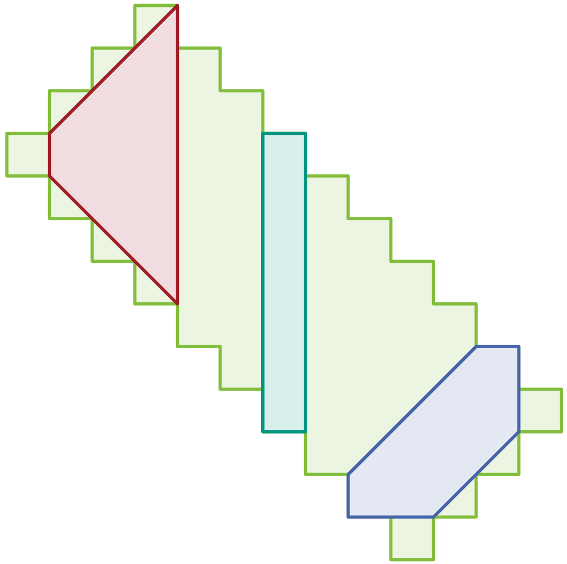
Convex/Reflex Vertices

Maximal clique in $G_P \leftrightarrow$ **maximal** convex subpolygon on vertices of P.



Convex/Reflex Vertices

Maximal clique in $G_P \leftrightarrow$ **maximal** convex subpolygon on vertices of P.



Department of Computer Science
Colgate University

Convex/Reflex Vertices

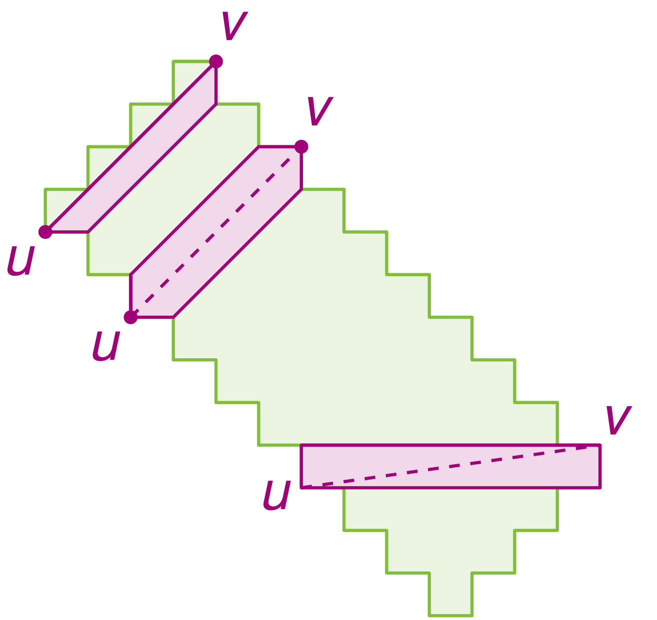
Maximal clique in $G_P \leftrightarrow$ **maximal** convex subpolygon on vertices of P.

Structural lemma:

Every convex vertex u has a convex neighbor v, such that (u, v) is in one maximal clique.

and

Edges incident to a reflex vertex are in two or more maximal cliques



Convex/Reflex Vertices

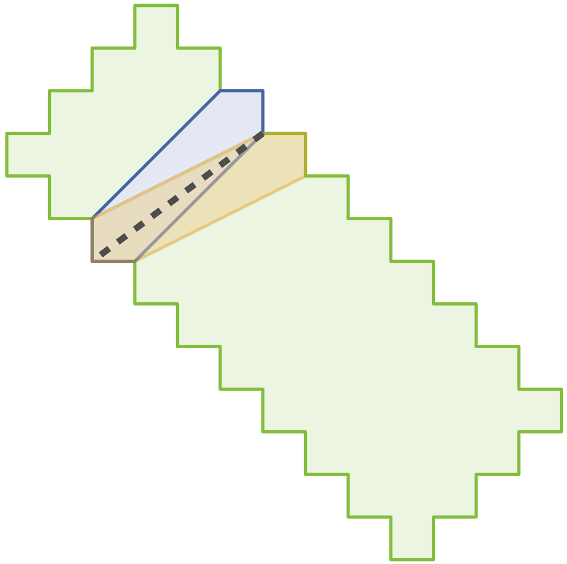
Maximal clique in $G_P \leftrightarrow$ **maximal** convex subpolygon on vertices of P.

Structural lemma:

Every convex vertex u has a convex neighbor v, such that (u, v) is in one maximal clique.

and

Edges incident to a reflex vertex are in two or more maximal cliques



Convex/Reflex Vertices

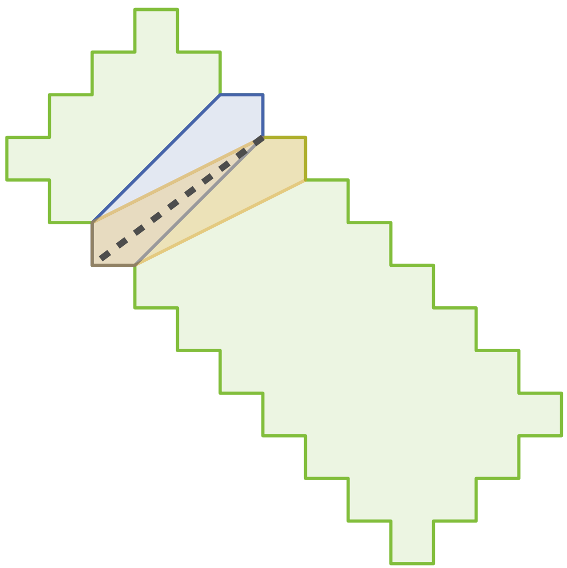
Maximal clique in $G_P \leftrightarrow$ **maximal** convex subpolygon on vertices of P.

Structural lemma:

Every convex vertex u has a convex neighbor v, such that (u, v) is in one maximal clique.

and

Edges incident to a reflex vertex are in two or more maximal cliques



Now Simple:

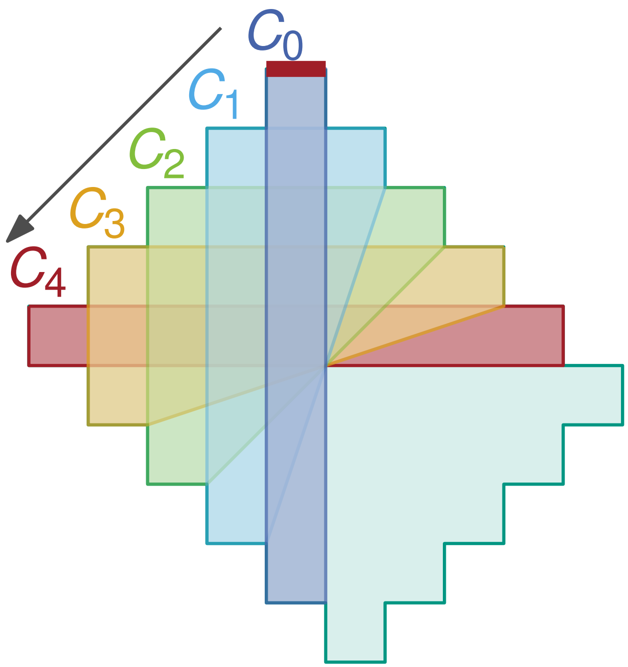
Compute all convex vertices by computing all edges contained in exactly one maximal clique.

Initial staircase reconstruction

Elementary clique:

A maximal clique containing 3 convex vertices*.

Elementary cliques can be **ordered**, starting from a tab.



Initial staircase reconstruction

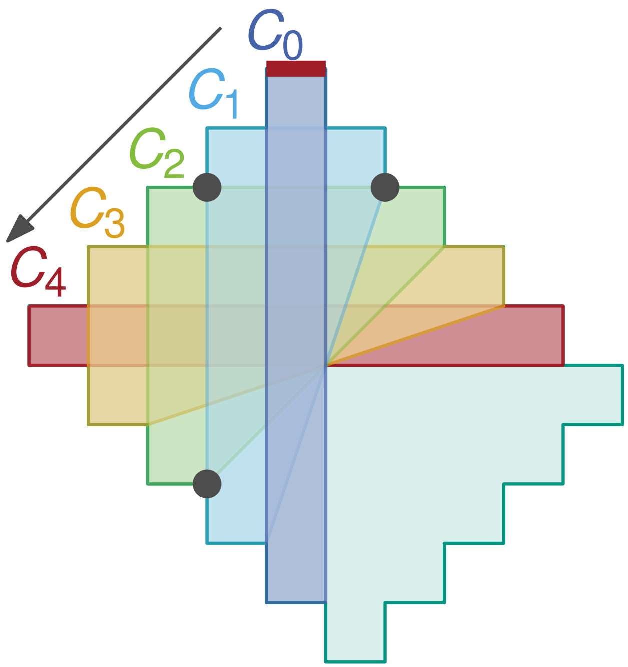
Elementary clique:

A maximal clique containing 3 convex vertices*.

Elementary cliques can be **ordered**, starting from a tab.

C_i shares 3 reflex vertices with C_{i-1}

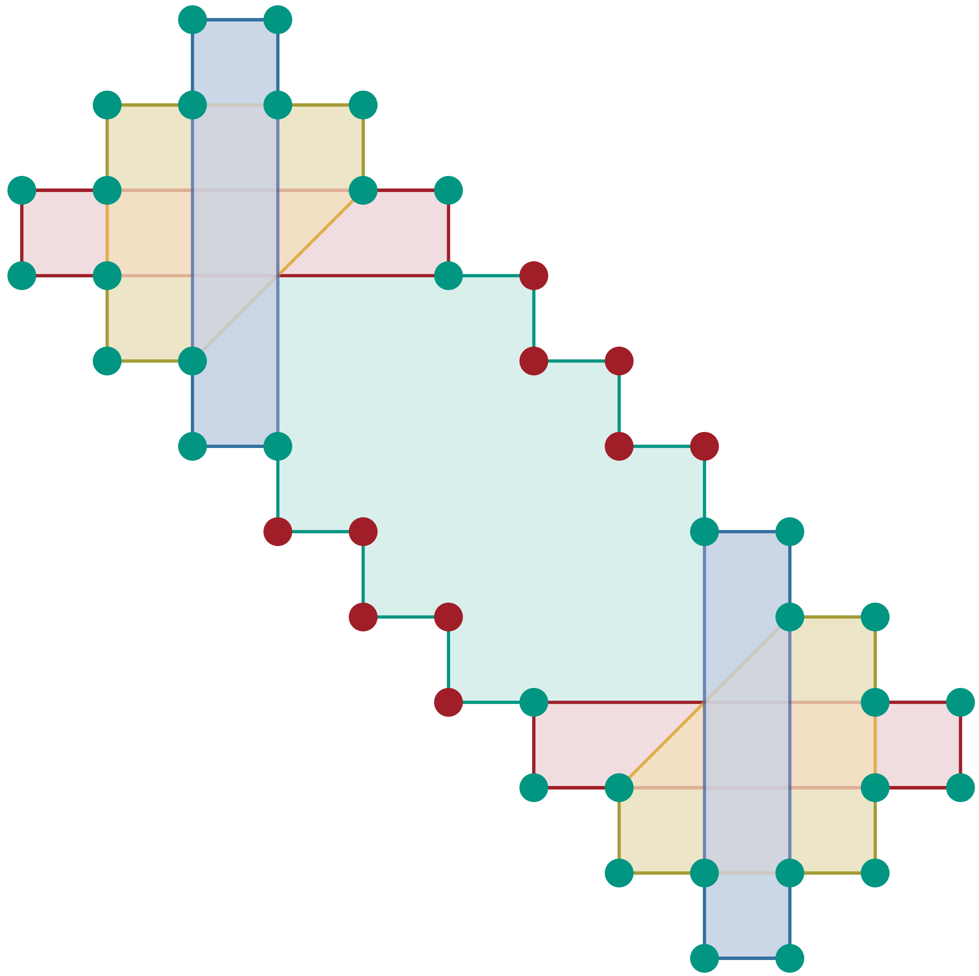
$\rightarrow$ Orders the clique vertices along **some** staircase.



Filling in remaining staircases

We know some vertices...

But some vertices remain after looking at all elementary cliques.

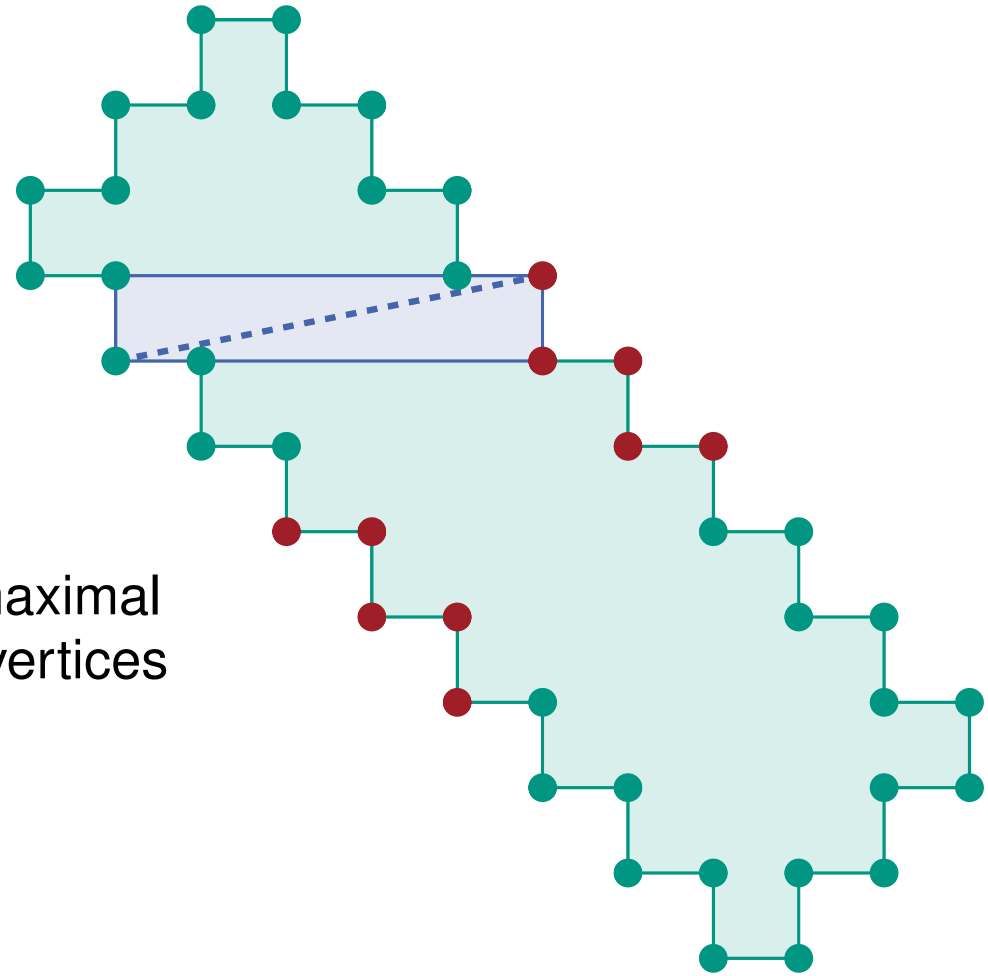


Filling in remaining staircases

We know some vertices...

But some vertices remain after looking at all elementary cliques.

Construct via “rectangular” maximal cliques from **known** convex vertices to **unknown** convex vertices.

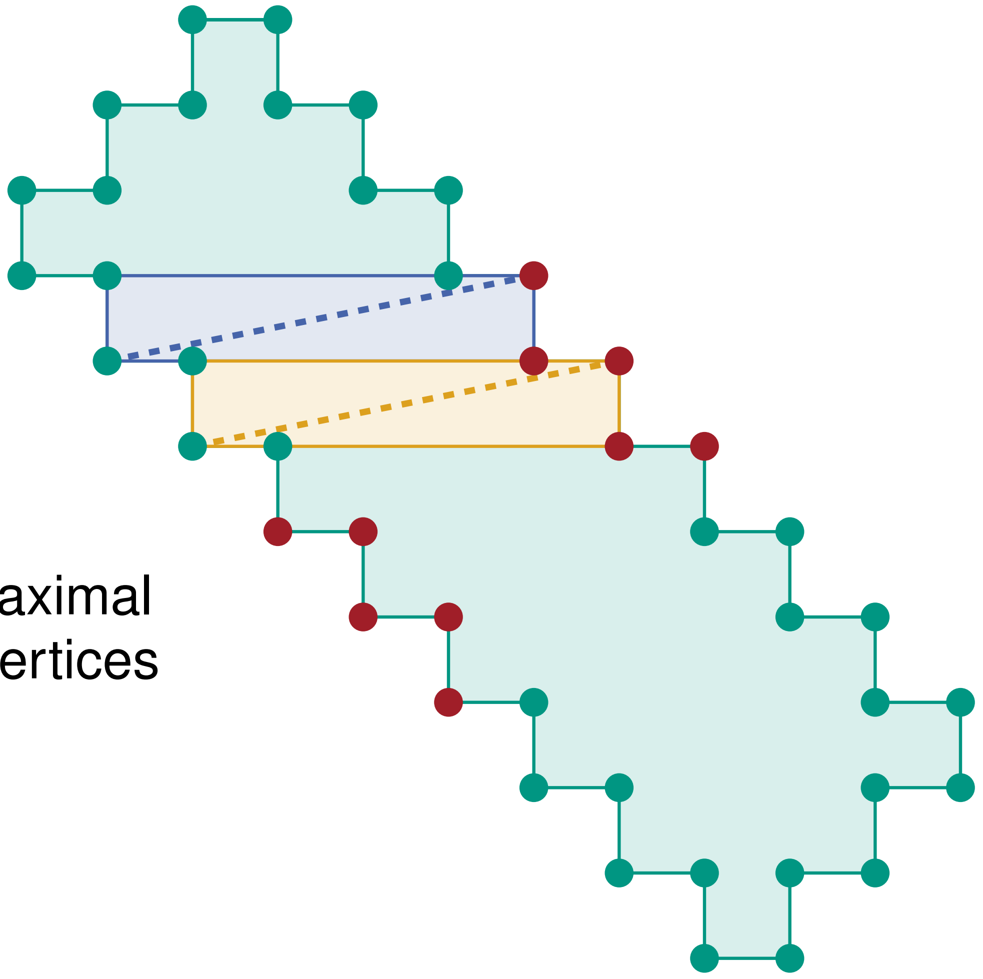


Filling in remaining staircases

We know some vertices...

But some vertices remain after looking at all elementary cliques.

Construct via “rectangular” maximal cliques from **known** convex vertices to **unknown** convex vertices.



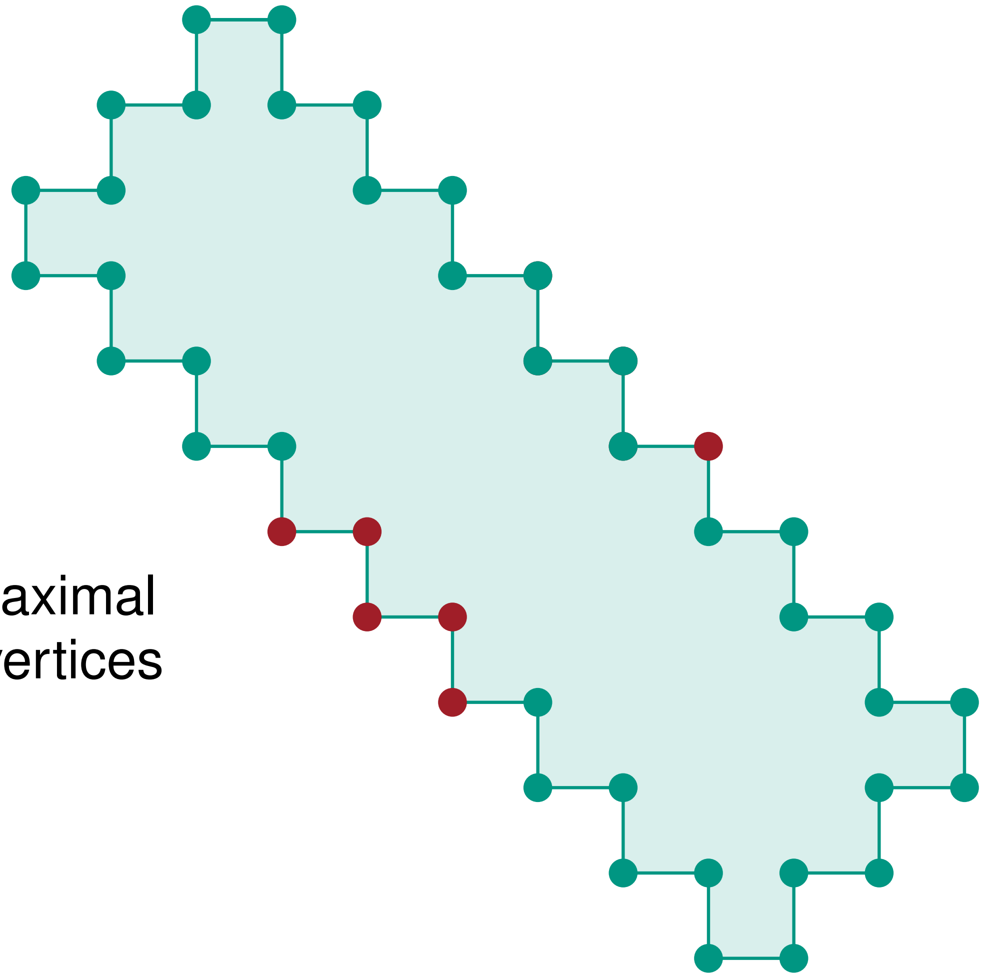
Department of Computer Science
Colgate University

Filling in remaining staircases

We know some vertices...

But some vertices remain after looking at all elementary cliques.

Construct via “rectangular” maximal cliques from **known** convex vertices to **unknown** convex vertices.



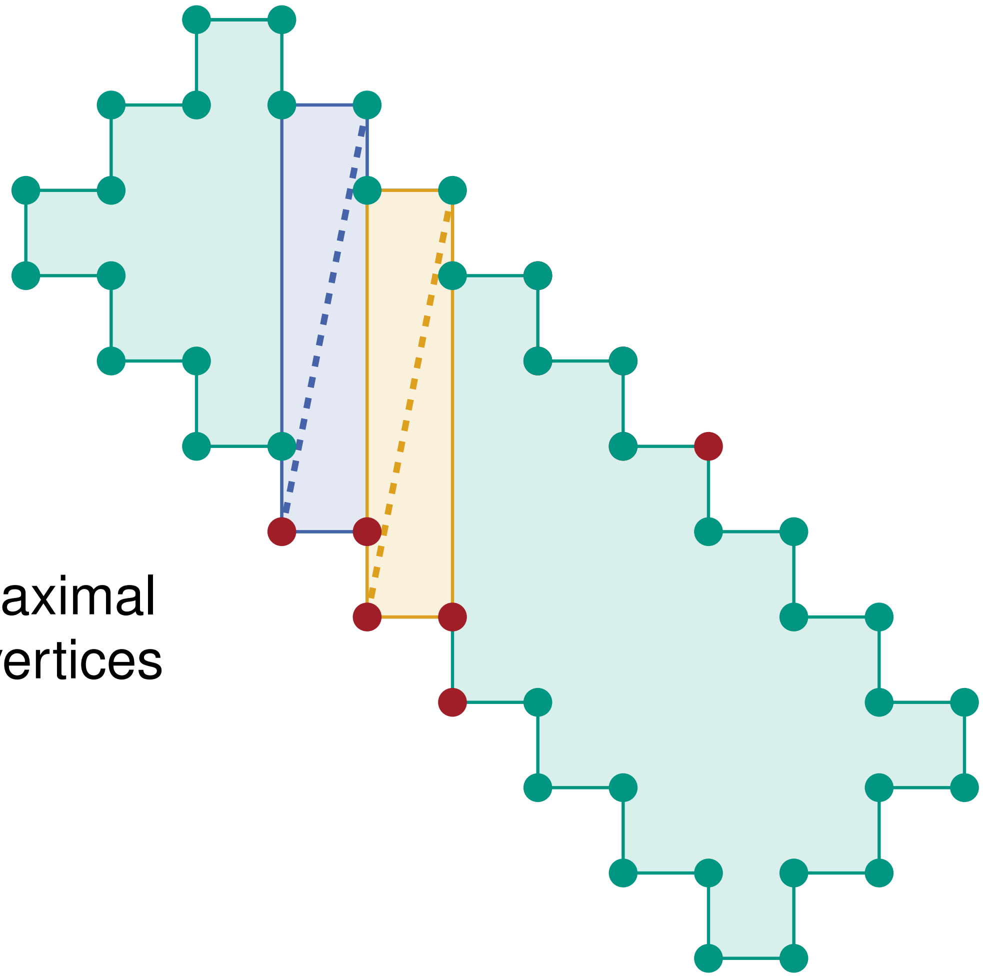
Department of Computer Science
Colgate University

Filling in remaining staircases

We know some vertices...

But some vertices remain after looking at all elementary cliques.

Construct via “rectangular” maximal cliques from **known** convex vertices to **unknown** convex vertices.

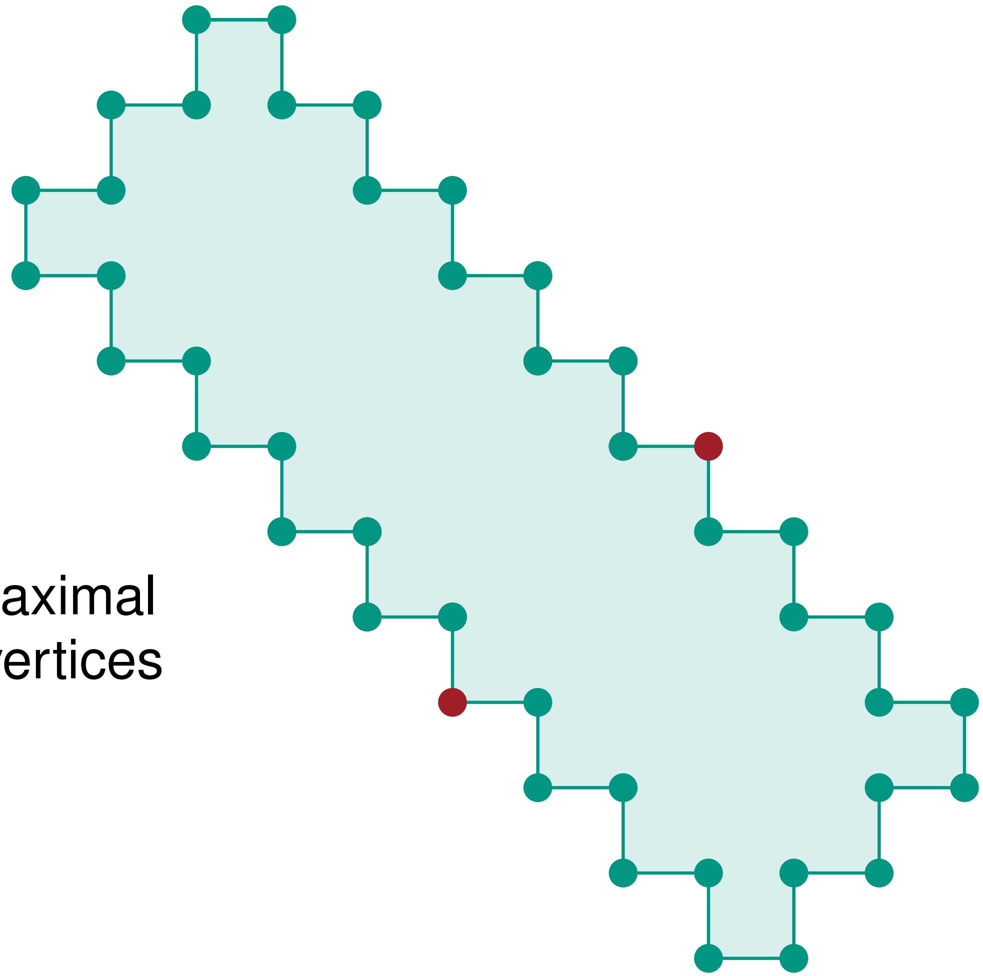


Filling in remaining staircases

We know some vertices...

But some vertices remain after looking at all elementary cliques.

Construct via “rectangular” maximal cliques from **known** convex vertices to **unknown** convex vertices.



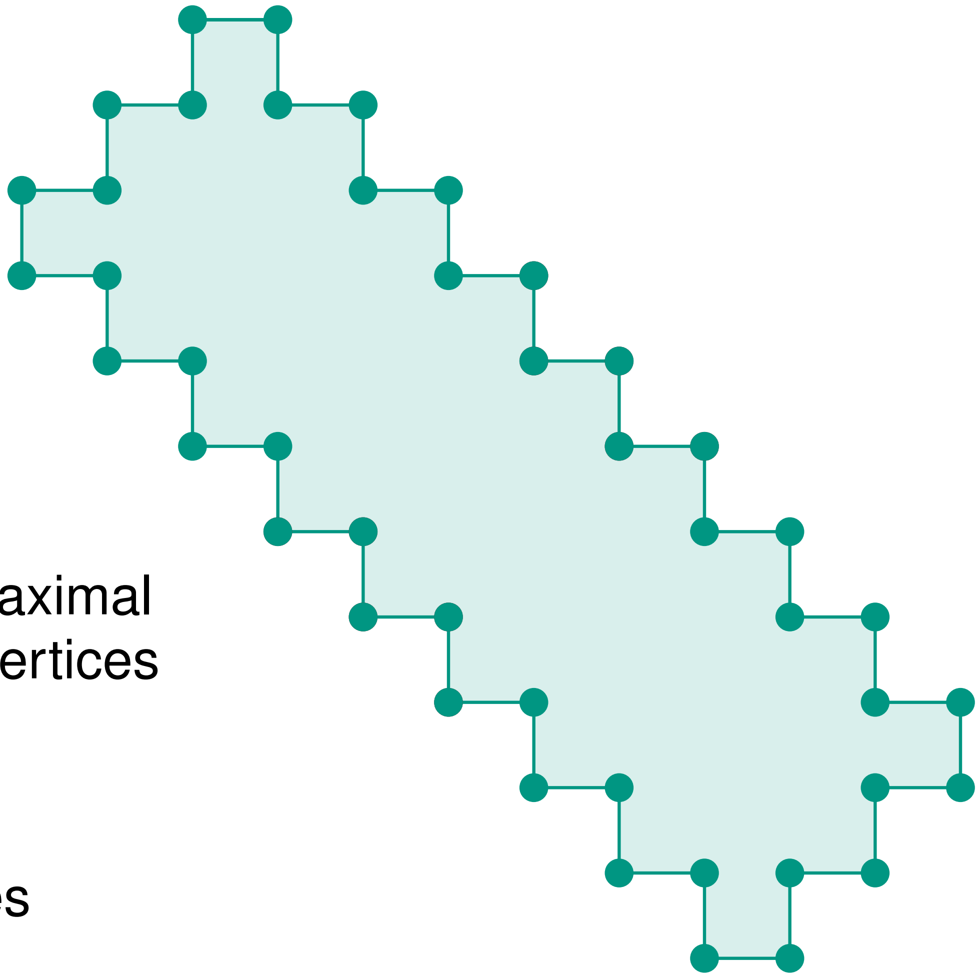
Filling in remaining staircases

We know some vertices...

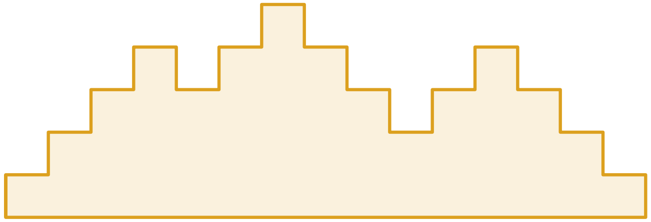
But some vertices remain after looking at all elementary cliques.

Construct via “rectangular” maximal cliques from **known** convex vertices to **unknown** convex vertices.

Gives us all remaining vertices

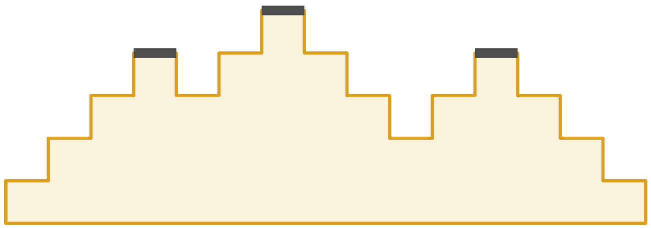


Histograms: high-level algorithm



Histograms: high-level algorithm

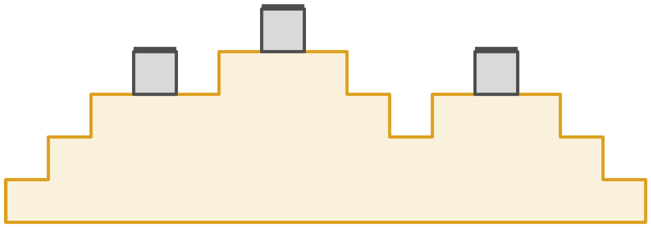
Step 1: Determine tabs



Histograms: high-level algorithm

Step 1: Determine tabs

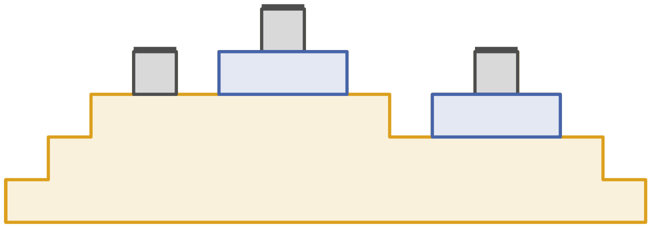
Step 2: Remove and repeat



Histograms: high-level algorithm

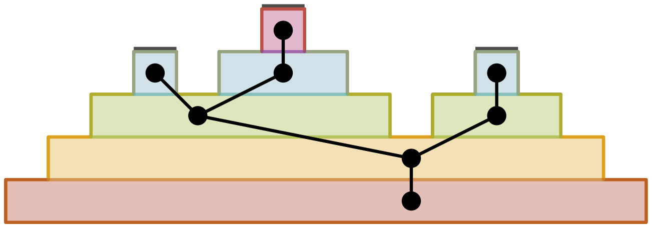
Step 1: Determine tabs

Step 2: Remove and repeat



Histograms: high-level algorithm

Step 1: Determine tabs

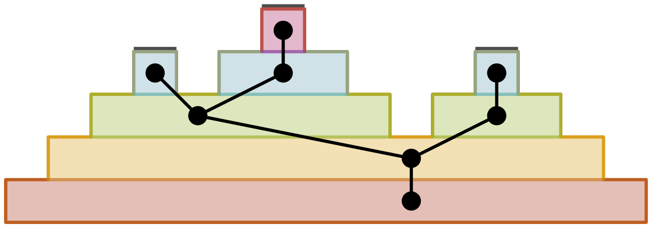


Step 2: Remove and repeat

Step 3: Construct a contract tree of all rectangles

Histograms: high-level algorithm

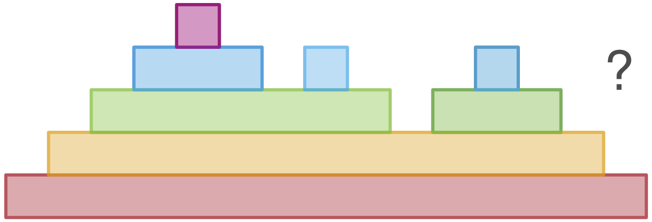
Step 1: Determine tabs

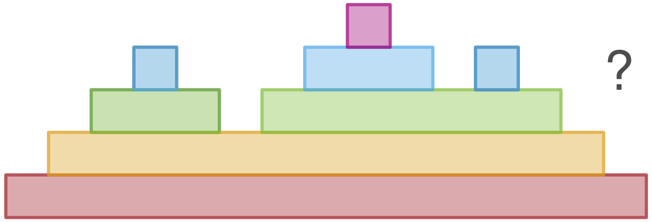


Step 2: Remove and repeat

Step 3: Construct a contract tree of all rectangles

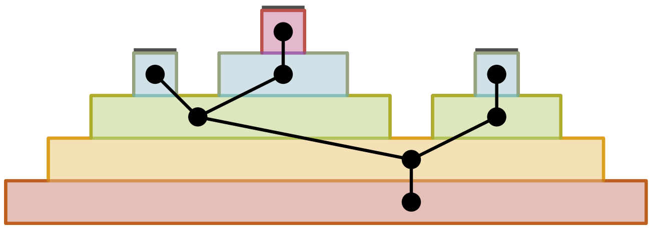
Step 4: Order the tree to construct the polygon





Histograms: high-level algorithm

Step 1: Determine tabs

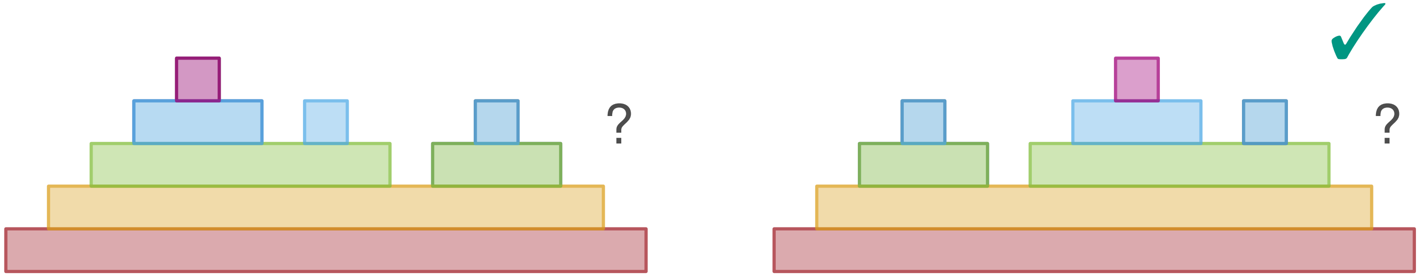


Step 2: Remove and repeat

Step 3: Construct a contract tree of all rectangles

Step 4: Order the tree to construct the polygon

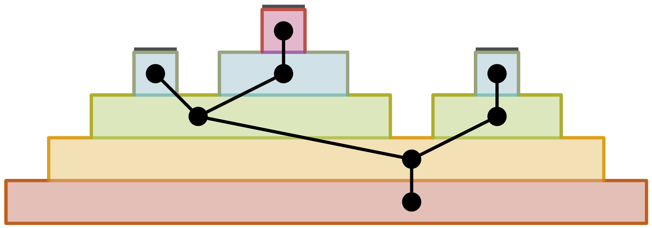
$\rightarrow$ generate polygon and check visibility graph



Histograms: Running time

An ordering of the leaves gives us an ordering of the tree

k leaves

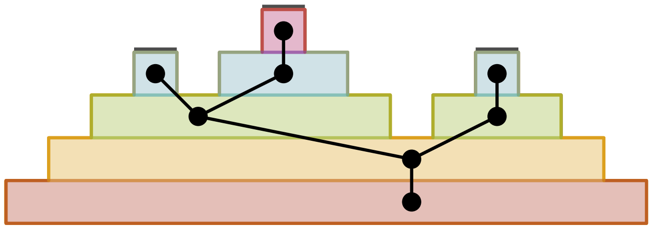


Histograms: Running time

An ordering of the leaves gives us an ordering of the tree

$\rightarrow$ and *maps* vertices to coordinates

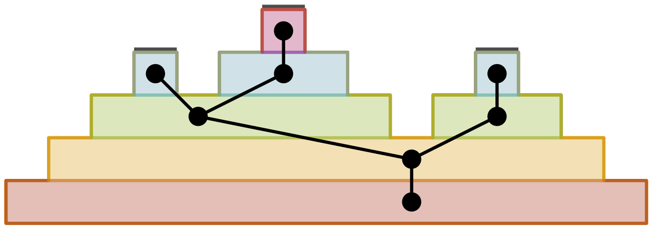
k leaves



Histograms: Running time

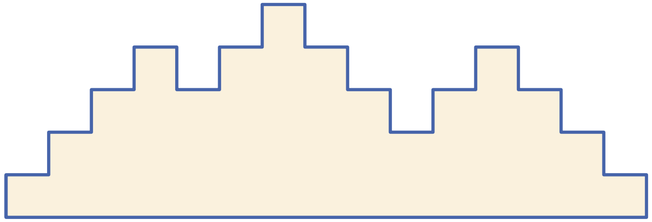
An ordering of the leaves gives us an ordering of the tree

k leaves



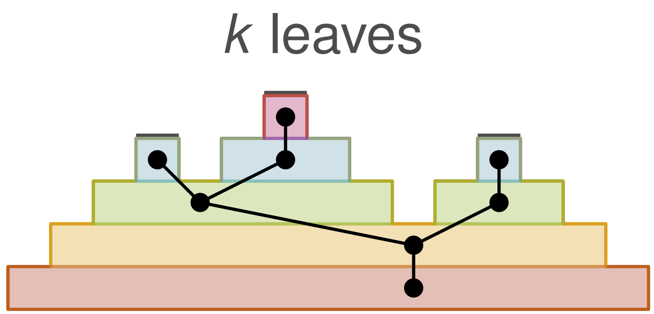
$\rightarrow$ and *maps* vertices to coordinates

Each ordering gives a polygon



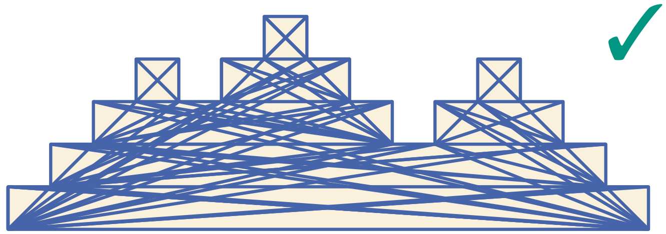
Histograms: Running time

An ordering of the leaves gives us an ordering of the tree



$\rightarrow$ and *maps* vertices to coordinates

Each ordering gives a polygon

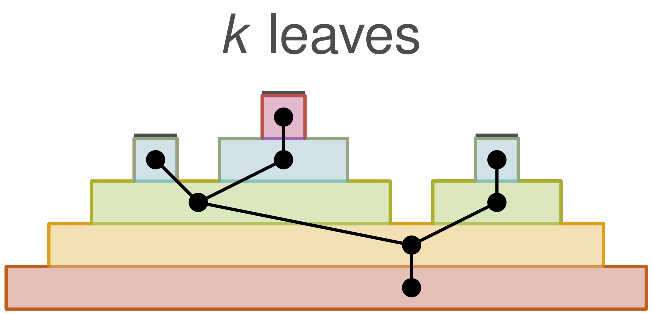


Compute its visibility graph and check for a match in $O(n \log n + m)$ time.

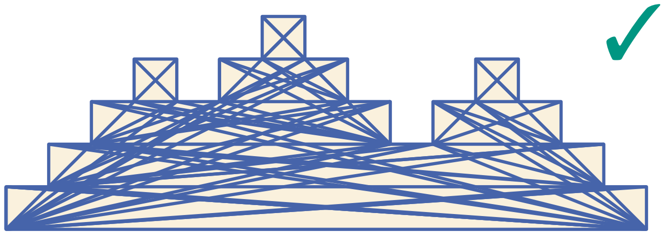
Histograms: Running time

An ordering of the leaves gives us an ordering of the tree

$\rightarrow$ and *maps* vertices to coordinates



Each ordering gives a polygon



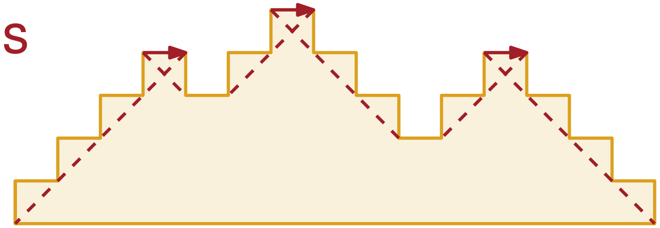
Compute its visibility graph and check for a match in $O(n \log n + m)$ time.

Seems like $O(k!(n \log n + m))$ time...

Histograms: Running time

But we also need to *orient* the tabs from left to right.

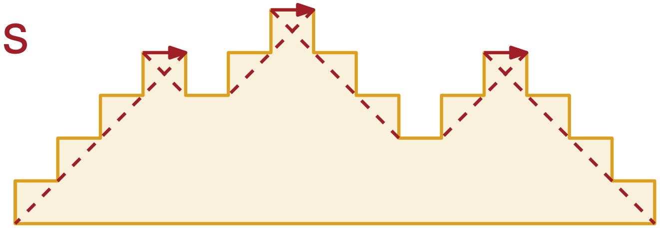
$\rightarrow$ 2^k possible orientations of leaves



Histograms: Running time

But we also need to *orient* the tabs from left to right.

$\rightarrow$ 2^k possible orientations of leaves



+ time to compute 1-simplicial edges

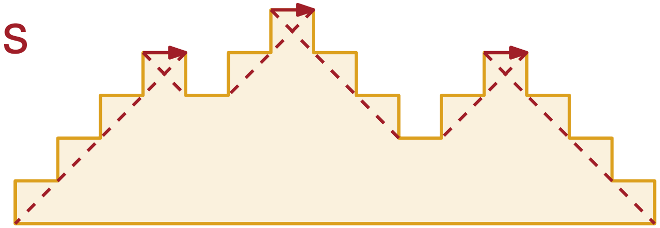
$\rightarrow$ $O(n^2 m + k!2^k(n \log n + m))$ time.

Department of Computer Science
Colgate University

Histograms: Running time

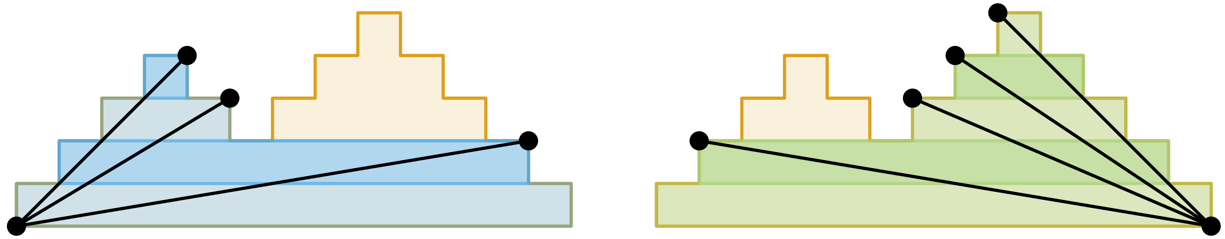
But we also need to *orient* the tabs from left to right.

$\rightarrow 2^k$ possible orientations of leaves



+ time to compute 1-simplicial edges

$\rightarrow O(n^2 m + k!2^k(n \log n + m))$ time.



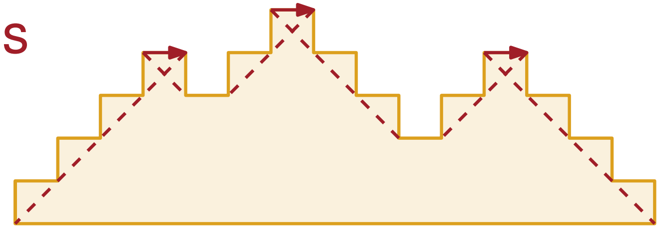
Can fix the outer two paths, leaving us with only $(k-2)!2^{k-2}$ options

$\rightarrow O(n^2 m + (k-2)!2^{k-2}(n \log n + m))$ time.

Histograms: Running time

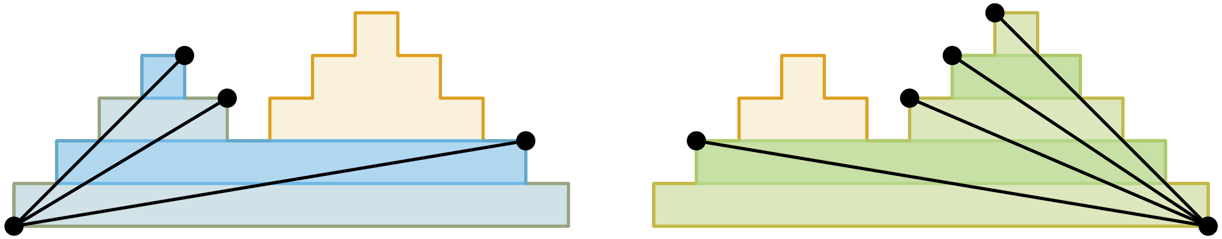
But we also need to *orient* the tabs from left to right.

$\rightarrow$ 2^k possible orientations of leaves



+ time to compute 1-simplicial edges

$\rightarrow$ $O(n^2 m + k!2^k(n \log n + m))$ time.



Can fix the outer two paths, leaving us with only $(k-2)!2^{k-2}$ options

$\rightarrow$ $O(n^2 m + (k-2)!2^{k-2}(n \log n + m))$ time.

$\rightarrow$ $O(n^2 m)$ time for binary trees (recursively fix right/left spine)

Nodari Sitchinava and Darren Strash:
Reconstructing Generalized Staircase Polygons with Uniform Step Length

Department of Computer Science
Colgate University

Conclusion

Since we assign vertices to coordinates, we get **recognition** for free.

$\rightarrow$ compute coordinates and check visibility graph

Conclusion

Since we assign vertices to coordinates, we get **recognition** for free.

$\rightarrow$ compute coordinates and check visibility graph

General problem is still open...

- What about orthogonal polygons with fewer restrictions?
- Is it possible to reconstruct orthogonal convex fans in polynomial time?
- Are there more general classes of polygons that can be recognized / reconstructed?

Conclusion

Since we assign vertices to coordinates, we get **recognition** for free.

$\rightarrow$ compute coordinates and check visibility graph

General problem is still open...

- What about orthogonal polygons with fewer restrictions?
- Is it possible to reconstruct orthogonal convex fans in polynomial time?
- Are there more general classes of polygons that can be recognized / reconstructed?

Thank You!