Rectangular Partitioning of Rectilinear Polygons

Exercise 15

Alexandre Ros

November 4, 2023

Introduction to the problem

Exercise 15

Decompose a rectilinear polygon into rectangles, using segments aligned with the edges of the polygon. The algorithm must produce the **minimum number of pieces**, and the output must give a complete description of the partition.

Input

A polygon whose sides meet at right angles.

Output

A **partition** of the polygon into rectangles, with no overlaps.

We will call this problem *MNC* (minimal nonoverlapping cover).

Examples

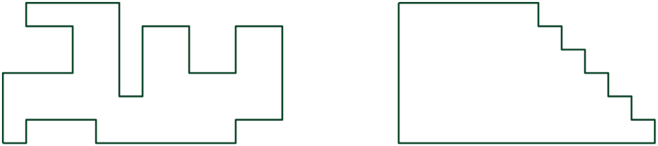


Figure 1: Two rectilinear polygons.

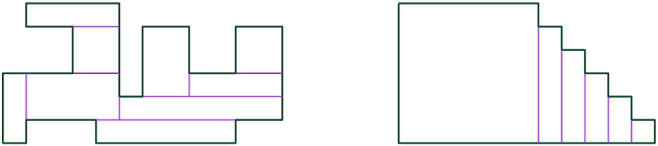


Figure 2: A minimum partition of both.

Preliminaries

Concave and convex vertices

In a rectilinear polygon, we can distinguish between concave and convex vertices.

- A vertex is said to be **concave** if its interior angle is over 180°.
- A vertex is said to be **convex** if its interior angle is under 180°.

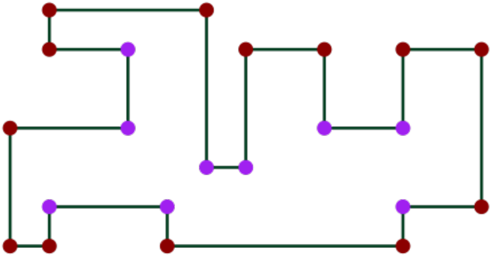


Figure 3: Convex (red) and concave (purple) vertices.

Co-vertical, co-horizontal and chords

- Two vertices (x_1, y_1), (x_2, y_2) that share no edge are **co-vertical** $\iff y_1 = y_2$ and **co-horizontal** $\iff x_1 = x_2$.
- A **chord** is a segment fully contained inside the polygon that connects two co-horizontal or co-vertical vertices.

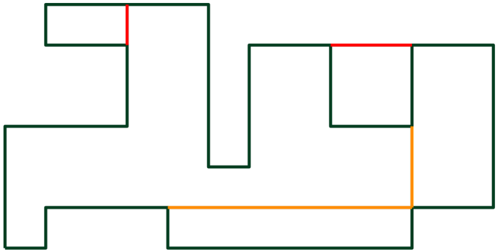


Figure 4: Examples of chords (orange) and **not** chords (red).

Chord-less polygons

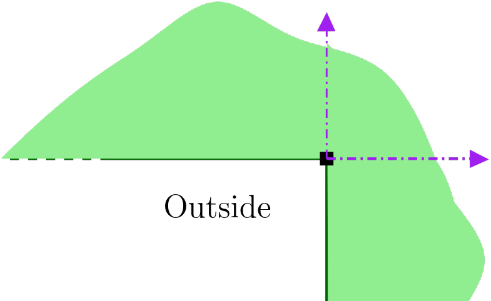
Suppose, for now, that there are no chords in polygon P.

Observation #1

P is a rectangle $\iff$ P has exactly four convex vertices.

Observation #2

A concave vertex must be a vertex of two rectangles of the partition. See example below.



Algorithm for chord-less polygons

Observation #3

If P is not a rectangle then it must have at least one concave vertex.

Idea to solve the problem

For each concave vertex, extend its vertical edge until it hits another edge. *If it hits a vertex, then that extension is a chord!*

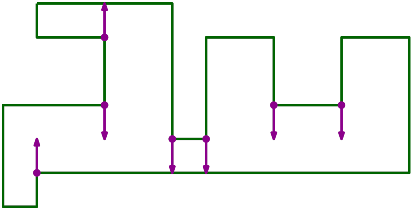


Figure 6: Chord-less polygons optimal solution.

Algorithm for chord-less polygons

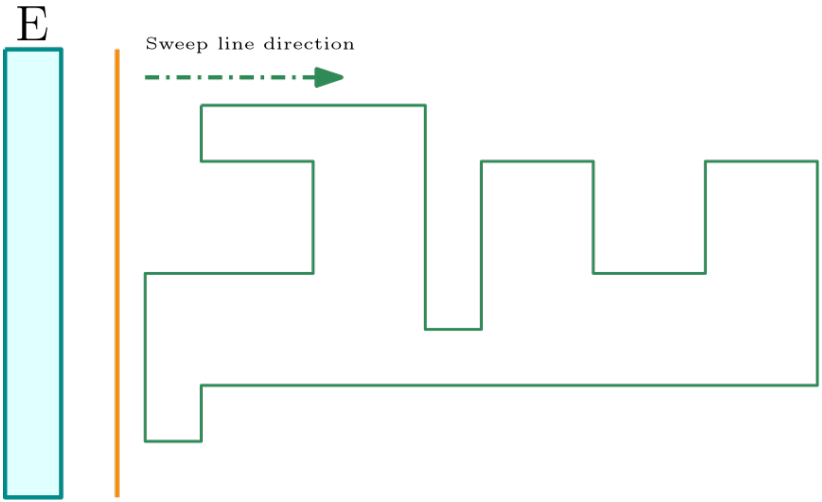
Algorithm 1 Chord-less MNC in $O(n \log n)$

procedure $\mathrm{MNC}(P)$

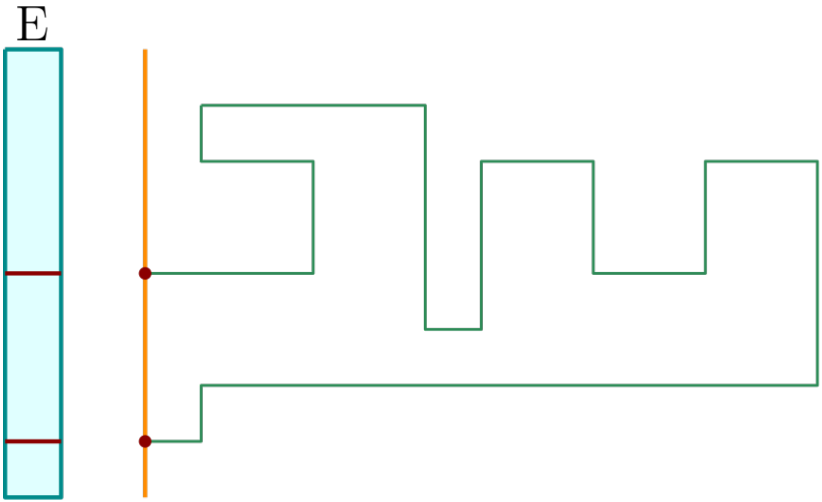
- $E \leftarrow \emptyset$; $R \leftarrow \emptyset$
- $L \leftarrow$ sorted vertices of P by x-coordinate[1].
- **for** each vertex v in P **do**
  - Insert v to E in position, delete position if duplicate.
  - **if** v is concave **then**
    - Extend vertical edge.
    - Insert to R the vertical edge.
- Return R as the paritioning edges of P.

[1] If using y-coordinate as tiebreaker, we can read pairs of vertices at a time (vertical edges).

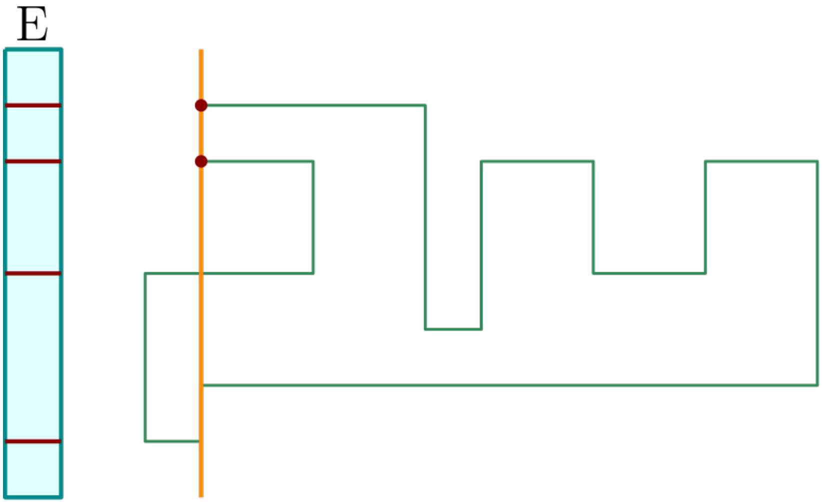
Algorithm for chord-less polygons

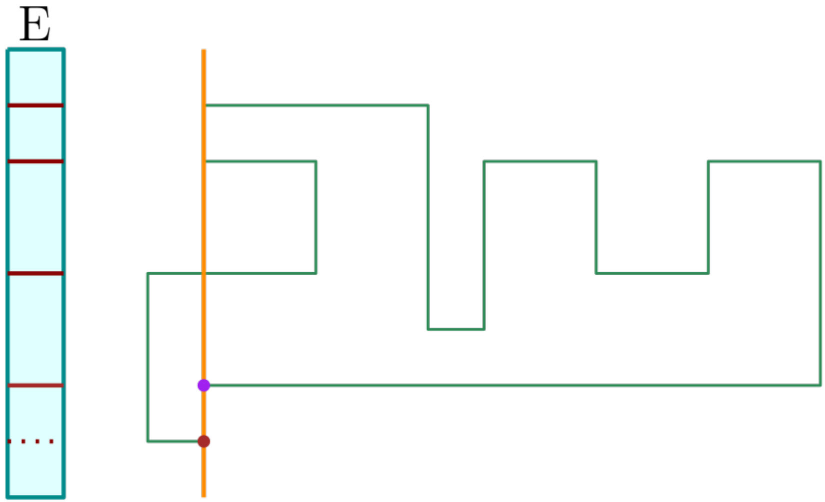


Algorithm for chord-less polygons

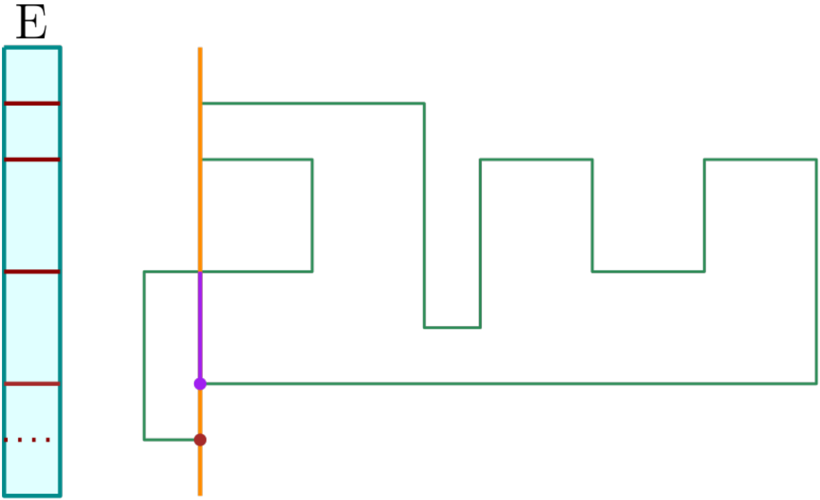


Algorithm for chord-less polygons

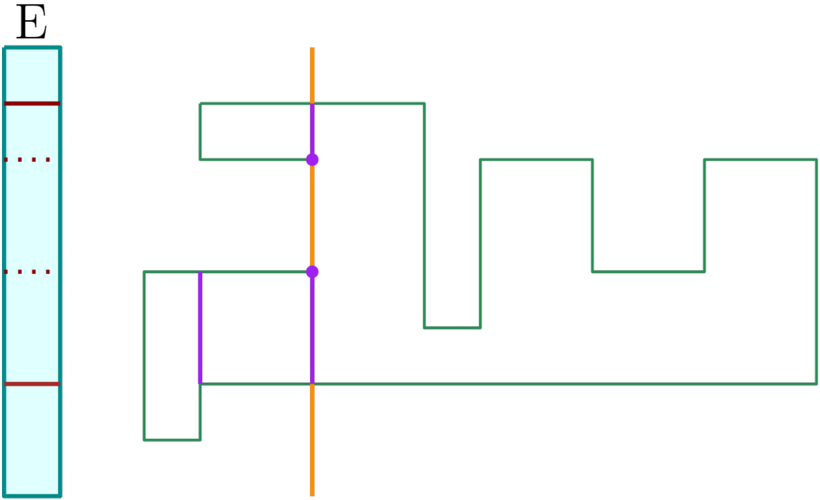



E

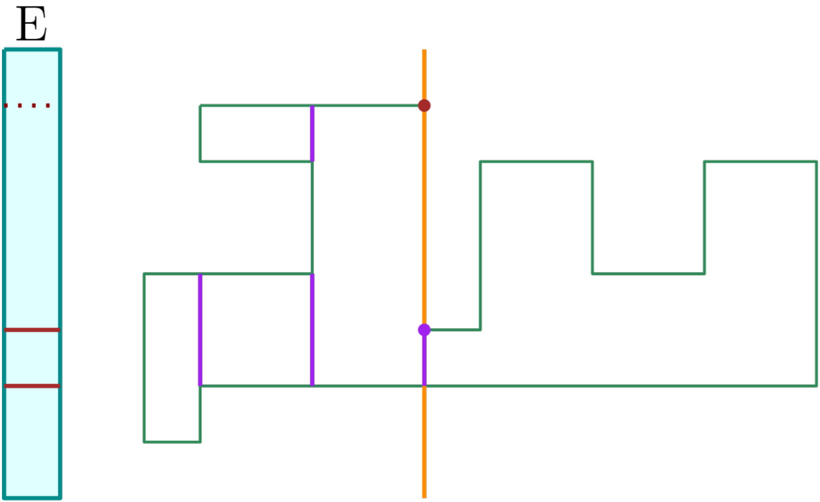
Algorithm for chord-less polygons



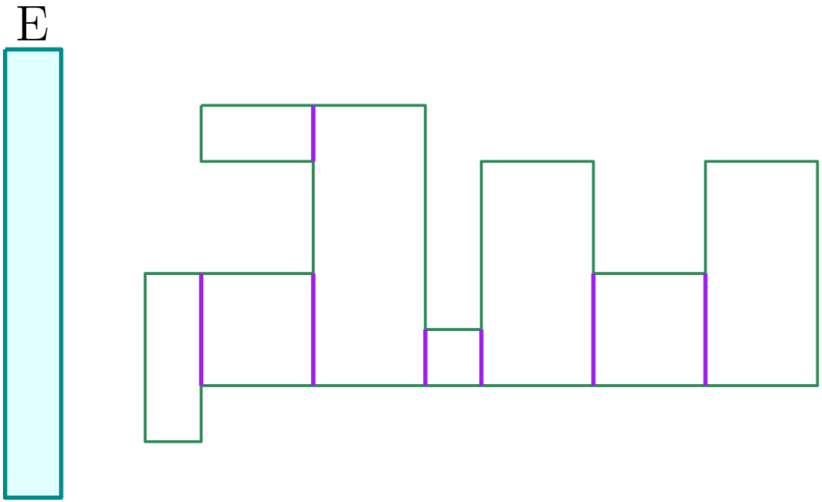
Algorithm for chord-less polygons



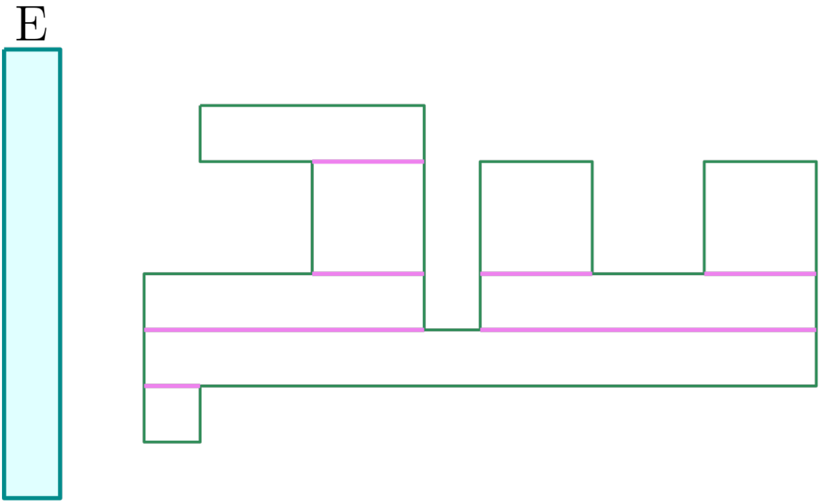
Algorithm for chord-less polygons



Algorithm for chord-less polygons



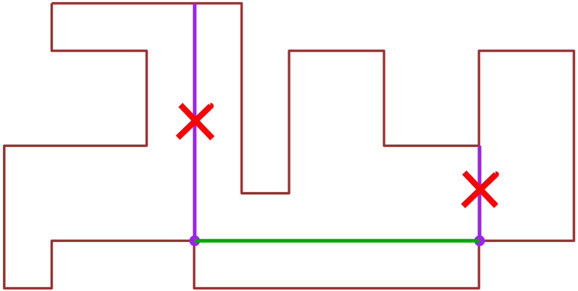
Algorithm for chord-less polygons



Polygons with chords

Problem #1

Suppose there are two co-horizontal vertices which form a chord. We may prefer joining that chord rather than extending the vertical edge.

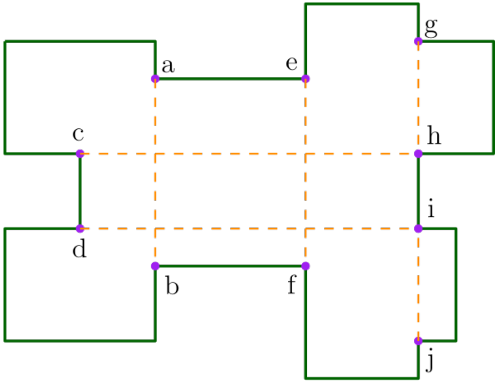


We would end up with one more rectangle!

Polygons with chords

Problem #2

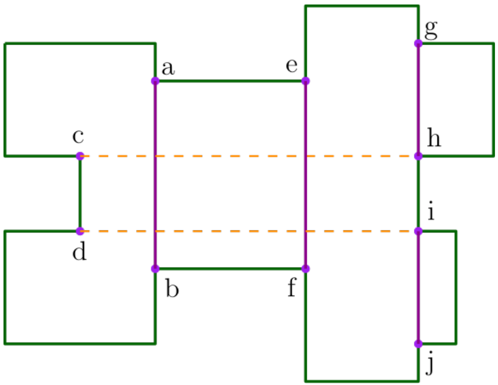
If we drew all chords, there may be intersections.



Chords *ab* and *ch* intersect. So do chords *ij* and *di*.

Solution

Draw the largest set of non-intersecting chords. After this step, no chords can remain. The remaining sub-polygons can be partitioned as usual.



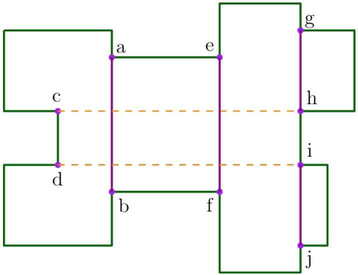
Polygons with chords

Theorem

A rectilinear polygon R has a minimum partition of order $N - L + 1$, where

$N =$ Total number of concave vertices on the boundary of R.

$L =$ Maximum number of nonintersecting chords.



$N = 10$ and $L = 4$ therefore the optimal order is 7.

Finding the largest set of nonintersecting chords

Process

Construct a bipartite graph, with vertices v_i for vertical chords, h_i for horizontal chords, and edges $\{v_i, h_j\}$ if v_i and h_j intersect.

Conclusion

Conclusion

- Summarize key points.
- Highlight takeaways.

Thank You

Thank you for your attention!