Computational Geometry-Introduction

Aritra Banik*
National Institute of Science Education and Research

Summer School on Graph Theory and Graph Algorithm at NIT Calicut

*Slide ideas borrowed from Marc van Kreveld and Subhash Suri

Outline

1. Introduction
2. Convex Hull
3. Art Gallery Problem

Computational geometry

- Study of geometric problems that arise in various applications and how geometric algorithms can help to solve well-defined versions of such problems
- Application areas that require geometric algorithms are computer graphics, motion planning and robotics, geographic information systems, CAD/CAM, statistics, physics simulations, databases, games, multimedia retrieval

Computational geometry history

EARLY 70S: First attention for geometric problems from algorithms researchers

1976: First PhD thesis in computational geometry (Michael Shamos)

1985: First Annual ACM Symposium on Computational Geometry. Also: first textbook

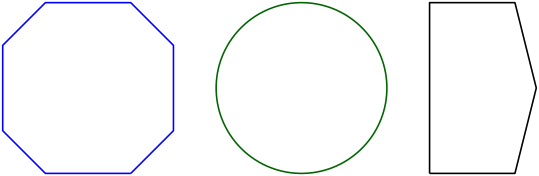
1996: CGAL: first serious implementation effort for robust geometric algorithms

1997: First handbook on computational geometry (second one in 2000)

Outline

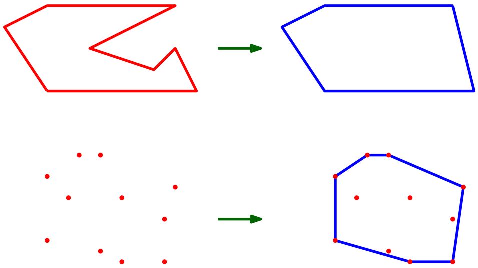
1. Introduction
2. Convex Hull
3. Art Gallery Problem

Convexity



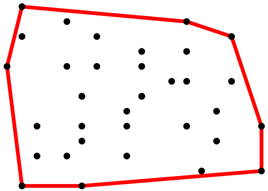
A shape or set is CONVEX if for any two points that are part of the shape, the whole connecting line segment is also part of the shape

Convex hull



For any subset of the plane (set of points, rectangle, simple polygon), its convex hull is the smallest CONVEX SET that contains that subset.

Convex hull problem

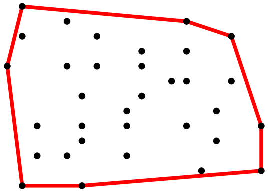


Give an algorithm that computes the convex hull of any given set of n points in the plane efficiently.

The input has $2n$ coordinates, so $O(n)$ size

For non-negative functions, $f(n)$ and $g(n)$, if there exists an integer n_0 and a constant $c > 0$ such that for all integers $n > n_0$, $f(n) \leq cg(n)$, then $f(n)$ is big O of $(g(n))$. This is denoted as $f(n) = O(g(n))$.

Convex hull problem



Give an algorithm that computes the convex hull of any given set of n points in the plane efficiently.

The input has $2n$ coordinates, so $O(n)$ size

For non-negative functions, $f(n)$ and $g(n)$, if there exists an integer n_0 and a constant $c > 0$ such that for all integers $n > n_0$, $f(n) \leq cg(n)$, then $f(n)$ is big O of $(g(n))$. This is denoted as $f(n) = O(g(n))$.

Convex hull problem

- PROPERTY The vertices of the convex hull are always points from the input
- Consequently, the edges of the convex hull connect two points of the input
- PROPERTY The supporting line of any convex hull edge has all input points to one side
- All points lie right of the directed line from p to q, if the edge from p to q is a CW convex hull edge.
- ALGORITHM?

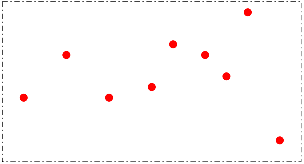
Convex hull problem

- PROPERTY The vertices of the convex hull are always points from the input
- Consequently, the edges of the convex hull connect two points of the input
- PROPERTY The supporting line of any convex hull edge has all input points to one side
- All points lie right of the directed line from p to q, if the edge from p to q is a CW convex hull edge.
- ALGORITHM?

Convex hull problem

- PROPERTY The vertices of the convex hull are always points from the input
- Consequently, the edges of the convex hull connect two points of the input
- PROPERTY The supporting line of any convex hull edge has all input points to one side
- All points lie right of the directed line from p to q, if the edge from p to q is a CW convex hull edge.
- ALGORITHM?

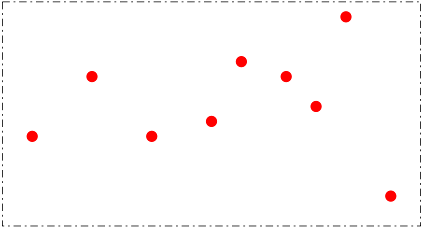
An incremental Algorithm



An incremental Algorithm

- Incremental, from left to right
- Let's first compute the upper boundary of the convex hull this way (property: on the upper hull, points appear in x-order)
- Main idea: Sort the points from left to right (= by x-coordinate).
- Then insert the points in this order, and maintain the upper hull so far

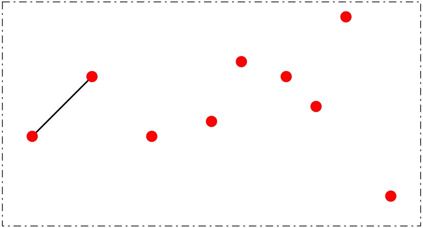
An incremental Algorithm



An incremental Algorithm

- Observation: from left to right, there are only right turns on the upper hull
- Initialize by inserting the leftmost two points
- If we add the third point there will be a right turn at the previous point, so we add it.
- If we add the fourth point we get a left turn at the third point
- So we remove the third point from the upper hull when we add the fourth

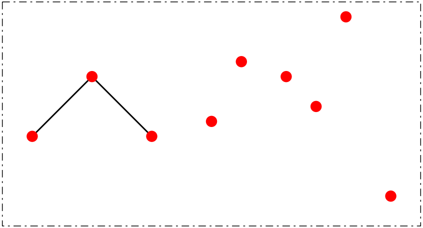
An incremental Algorithm



An incremental Algorithm

- Observation: from left to right, there are only right turns on the upper hull
- Initialize by inserting the leftmost two points
- If we add the third point there will be a right turn at the previous point, so we add it.
- If we add the fourth point we get a left turn at the third point
- So we remove the third point from the upper hull when we add the fourth

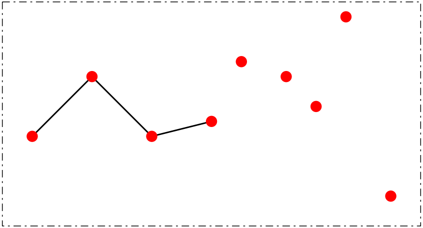
An incremental Algorithm



An incremental Algorithm

- Observation: from left to right, there are only right turns on the upper hull
- Initialize by inserting the leftmost two points
- If we add the third point there will be a right turn at the previous point, so we add it.
- If we add the fourth point we get a left turn at the third point
- So we remove the third point from the upper hull when we add the fourth

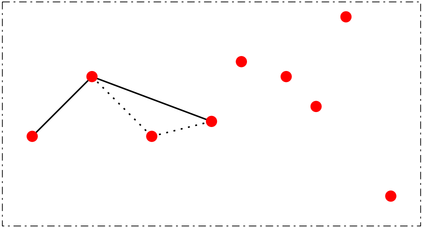
An incremental Algorithm



An incremental Algorithm

- Observation: from left to right, there are only right turns on the upper hull
- Initialize by inserting the leftmost two points
- If we add the third point there will be a right turn at the previous point, so we add it.
- If we add the fourth point we get a left turn at the third point
- So we remove the third point from the upper hull when we add the fourth

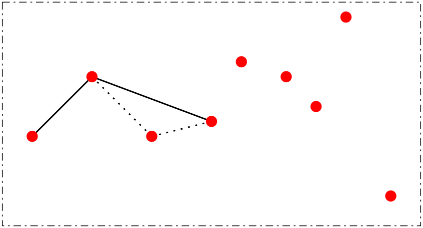
An incremental Algorithm



An incremental Algorithm

- Observation: from left to right, there are only right turns on the upper hull
- Initialize by inserting the leftmost two points
- If we add the third point there will be a right turn at the previous point, so we add it.
- If we add the fourth point we get a left turn at the third point
- So we remove the third point from the upper hull when we add the fourth

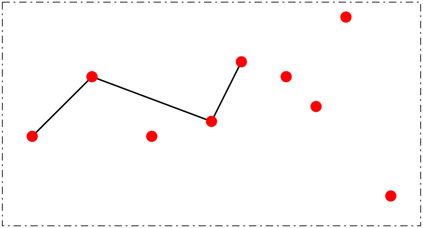
An incremental Algorithm



An incremental Algorithm

- If we add the fifth point we get a left turn at the fourth point
- So we remove the fourth point when we add the fifth
- and so on ...

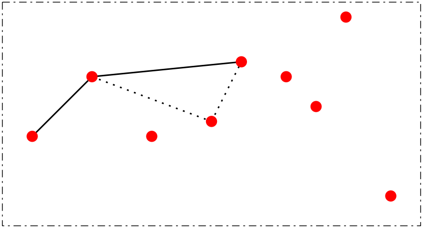
An incremental Algorithm



An incremental Algorithm

- If we add the fifth point we get a left turn at the fourth point
- So we remove the fourth point when we add the fifth
- and so on ...

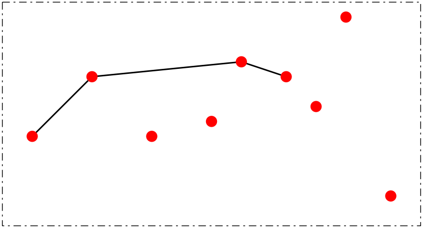
An incremental Algorithm



An incremental Algorithm

- If we add the fifth point we get a left turn at the fourth point
- So we remove the fourth point when we add the fifth
- and so on ...

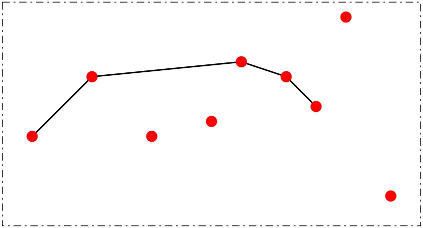
An incremental Algorithm



An incremental Algorithm

- If we add the fifth point we get a left turn at the fourth point
- So we remove the fourth point when we add the fifth
- and so on ...

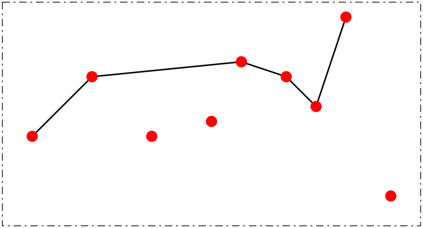
An incremental Algorithm



An incremental Algorithm

- If we add the fifth point we get a left turn at the fourth point
- So we remove the fourth point when we add the fifth
- and so on ...

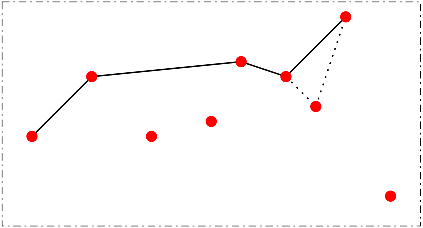
An incremental Algorithm



An incremental Algorithm

- If we add the fifth point we get a left turn at the fourth point
- So we remove the fourth point when we add the fifth
- and so on ...

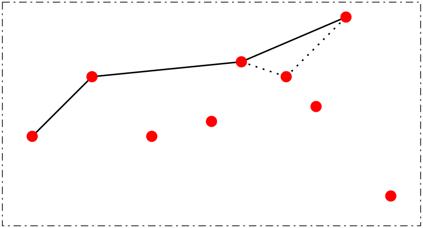
An incremental Algorithm



An incremental Algorithm

- If we add the fifth point we get a left turn at the fourth point
- So we remove the fourth point when we add the fifth
- and so on ...

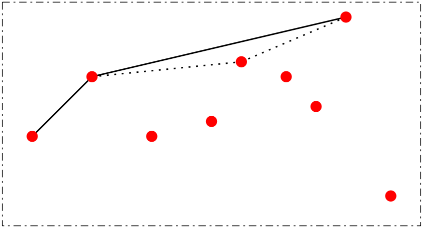
An incremental Algorithm



An incremental Algorithm

- If we add the fifth point we get a left turn at the fourth point
- So we remove the fourth point when we add the fifth
- and so on ...

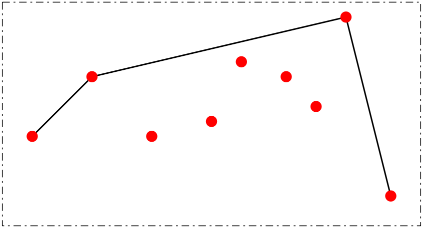
An incremental Algorithm



An incremental Algorithm

- If we add the fifth point we get a left turn at the fourth point
- So we remove the fourth point when we add the fifth
- and so on ...

An incremental Algorithm



An incremental Algorithm

- If we add the fifth point we get a left turn at the fourth point
- So we remove the fourth point when we add the fifth
- and so on ...

The pseudo-code

Algorithm 1 ConvexHull(P)

Input: A set P of points in the plane

Output: A list containing the vertices of $CH(P)$ in clockwise order

Sort the points by x-coordinate, resulting in a sequence $p_1.p_2, \cdots p_n$

Put the points p_1 and p_2 in a LIST UPPER, with p_1 as the first point.

for $i \leftarrow 3$ to n **do**

  Append p_i to LIST UPPER.

  while LIST UPPER contains more than two points and the last three points in LIST UPPER do not make a right turn **do**

    Delete the middle of the last three points from LIST UPPER

  end while

end for

Do the same for the lower convex hull, from right to left

Algorithm Analysis

Algorithm Analysis

- If we add the fifth point we get a left turn at the fourth point
- So we remove the fourth point when we add the fifth
- and so on ...

Algorithm Analysis

Algorithm Analysis

- If we add the fifth point we get a left turn at the fourth point
- So we remove the fourth point when we add the fifth
- and so on ...

Algorithm Analysis

Algorithm Analysis

- The sorting step takes $O(nlogn)$ time
- Adding a point takes $O(1)$ time for the adding-part. Removing points takes constant time for each removed point. If due to an addition, k points are removed, the step takes $O(1+k)$ time

Total time: $O(n \log n) + \sum_{3 \text{ to } n} O(1+k_i)$

Since Since $k_i = O(n)$, we get

Total time: $O(n^2)$

Global argument each point can be removed only once from the upper hull This gives us the fact:

$\sum_{3 \text{ to } n} k_i \leq n$

Total time: $O(n \log n)$

Algorithm Analysis

ALGORITHM ANALYSIS

- The sorting step takes $O(nlogn)$ time
- Adding a point takes $O(1)$ time for the adding-part. Removing points takes constant time for each removed point. If due to an addition, k points are removed, the step takes $O(1+k)$ time

TOTAL TIME: $O(n\log n) + \sum_{3 \text{ to } n} O(1+k_i)$
Since Since $k_i = O(n)$, we get
TOTAL TIME: $O(n^2)$
GLOBAL ARGUMENT each point can be removed only once from the upper hull This gives us the fact:
$\sum_{3 \text{ to } n} k_i \leq n$
TOTAL TIME: $O(n\log n)$

Algorithm Analysis

Algorithm Analysis

- The sorting step takes $O(nlogn)$ time
- Adding a point takes $O(1)$ time for the adding-part. Removing points takes constant time for each removed point. If due to an addition, k points are removed, the step takes $O(1+k)$ time

Total time: $O(n \log n) + \sum_{3 \text{ to } n} O(1+k_i)$
Since Since $k_i = O(n)$, we get
Total time: $O(n^2)$
Global argument each point can be removed only once from the upper hull This gives us the fact:
$\sum_{3 \text{ to } n} k_i \leq n$
Total time: $O(n \log n)$

Algorithm Analysis

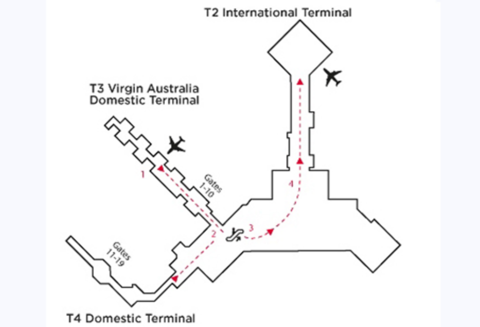
Algorithm Analysis

- The sorting step takes $O(nlogn)$ time
- Adding a point takes $O(1)$ time for the adding-part. Removing points takes constant time for each removed point. If due to an addition, k points are removed, the step takes $O(1+k)$ time

Total time: $O(n\log n) + \sum_{3\text{ to } n} O(1+k_i)$
Since Since $k_i = O(n)$, we get
Total time: $O(n^2)$
Global argument each point can be removed only once from the upper hull This gives us the fact:
$\sum_{3\text{ to } n} k_i \leq n$
Total time: $O(n\log n)$

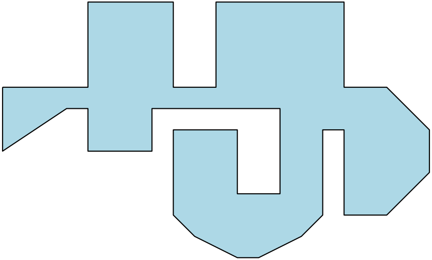
Outline

Art Gallery Theorem



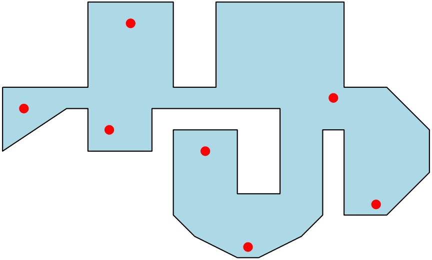
- The floor plan of an art gallery/museum/airport modeled as a simple polygon with n vertices.
- Objective is to secure the interior of the polygon by placing guards.
- Each guard is stationed at a fixed point, has 360°vision, and cannot see through the walls.
- How many guards needed to see the whole room?

Art Gallery Theorem



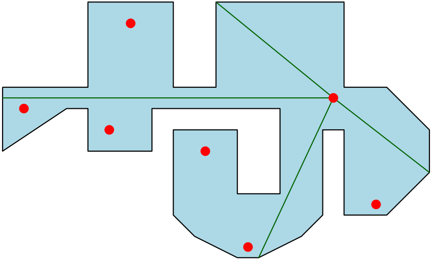
- The floor plan of an art gallery/museum/airport modeled as a simple polygon with n vertices.
- Objective is to secure the interior of the polygon by placing guards.
- Each guard is stationed at a fixed point, has 360°vision, and cannot see through the walls.
- How many guards needed to see the whole room?

Art Gallery Theorem



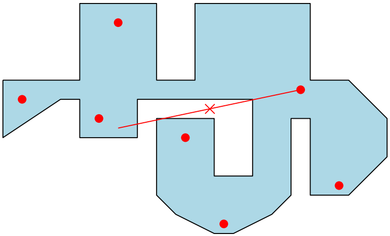
- The floor plan of an art gallery/museum/airport modeled as a simple polygon with n vertices.
- Objective is to secure the interior of the polygon by placing guards.
- Each guard is stationed at a fixed point, has 360°vision, and cannot see through the walls.
- How many guards needed to see the whole room?

Art Gallery Theorem



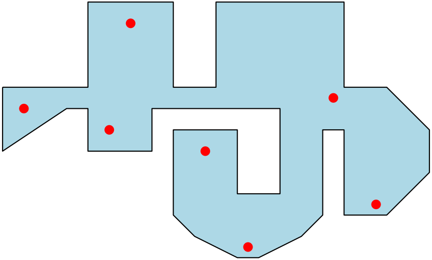
- The floor plan of an art gallery/museum/airport modeled as a simple polygon with n vertices.
- Objective is to secure the interior of the polygon by placing guards.
- Each guard is stationed at a fixed point, has 360°vision, and cannot see through the walls.
- How many guards needed to see the whole room?

Art Gallery Theorem



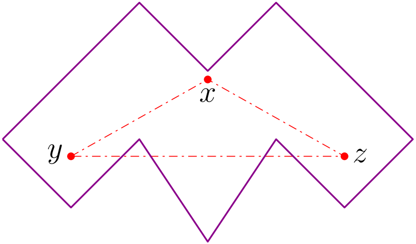
- The floor plan of an art gallery/museum/airport modeled as a simple polygon with n vertices.
- Objective is to secure the interior of the polygon by placing guards.
- Each guard is stationed at a fixed point, has 360°vision, and cannot see through the walls.
- How many guards needed to see the whole room?

Art Gallery Theorem



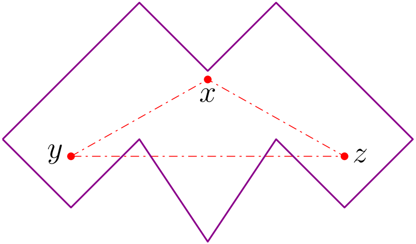
- The floor plan of an art gallery/museum/airport modeled as a simple polygon with n vertices.
- Objective is to secure the interior of the polygon by placing guards.
- Each guard is stationed at a fixed point, has 360°vision, and cannot see through the walls.
- How many guards needed to see the whole room?

Formulation



- Visibility: p, q visible if $pq \in P$.
- x is visible from y and z. But y and z not visible to each other.
- $g(P) =$ min. number of guards to see P
- $g(n) = \max_{|V(P)|=n} g(P)$ where maximum is taken over all simple polygons with n vertices
- Art Gallery Theorem asks for bounds on function $g(n)$: what is the smallest $g(n)$ that always works for any n-gon?

Formulation



- Visibility: p, q visible if $pq \in P$.
- x is visible from y and z. But y and z not visible to each other.
- $g(P) =$ min. number of guards to see P
- $g(n) = \max_{|V(P)|=n} g(P)$ where maximum is taken over all simple polygons with n vertices
- Art Gallery Theorem asks for bounds on function $g(n)$: what is the smallest $g(n)$ that always works for any n-gon?

Short story long:

- Problem posed to Vasek Chvatal by Victor Klee at a math conference in 1973. Chvatal solved it quickly with a complicated proof.
- Steve Fisk gave a proof from "THE BOOK".
- "THE BOOK" in which God keeps the most elegant proof of each mathematical theorem. During a lecture in 1985, Erdős said, "You don't have to believe in God, but you should believe in The Book."

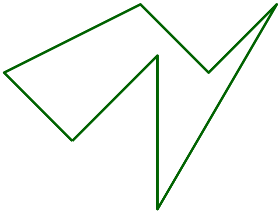
Short story long:

- Problem posed to Vasek Chvatal by Victor Klee at a math conference in 1973. Chvatal solved it quickly with a complicated proof.
- Steve Fisk gave a proof from "THE BOOK".
- "THE BOOK" in which God keeps the most elegant proof of each mathematical theorem. During a lecture in 1985, Erdős said, "You don't have to believe in God, but you should believe in The Book."

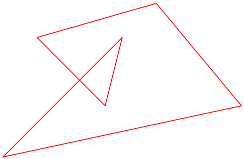
Short story long:

- Problem posed to Vasek Chvatal by Victor Klee at a math conference in 1973. Chvatal solved it quickly with a complicated proof.
- Steve Fisk gave a proof from "THE BOOK".
- "THE BOOK" in which God keeps the most elegant proof of each mathematical theorem. During a lecture in 1985, Erdős said, "You don't have to believe in God, but you should believe in The Book."

Simple Polygon



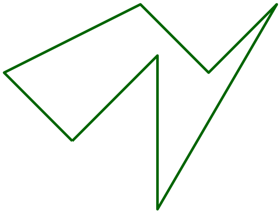
Simple Polygon



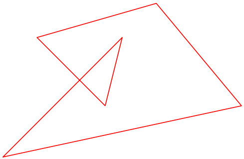
Not a simple polygon

- A simple polygon is a closed polygonal curve without self-intersection.
- By Jordan Theorem, a polygon divides the plane into interior, exterior, and boundary.
- We use polygon both for boundary and its interior; the context will make the usage clear.
- Polygons with holes are topologically different

Simple Polygon



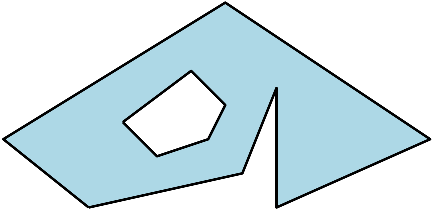
Simple Polygon



Not a simple polygon

- A simple polygon is a closed polygonal curve without self-intersection.
- By Jordan Theorem, a polygon divides the plane into interior, exterior, and boundary.
- We use polygon both for boundary and its interior; the context will make the usage clear.
- Polygons with holes are topologically different

Simple Polygon



- A simple polygon is a closed polygonal curve without self-intersection.
- By Jordan Theorem, a polygon divides the plane into interior, exterior, and boundary.
- We use polygon both for boundary and its interior; the context will make the usage clear.
- Polygons with holes are topologically different

Trying it Out

- $g(3)??g(4)??g(5)??$

- For $n = 3, 4, 5$, $g(n) = 1$
- Is there a general formula in terms of n?

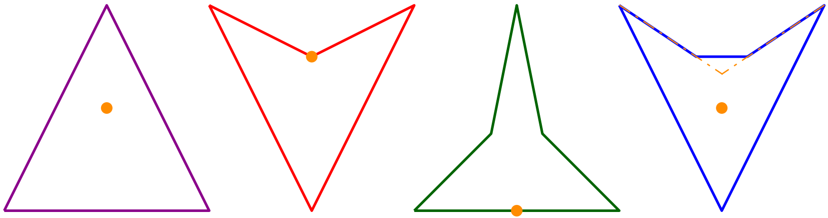
Trying it Out

- $g(3)??g(4)??g(5)??$

- For $n = 3, 4, 5,\ g(n) = 1$
- Is there a general formula in terms of n?

Trying it Out

- $g(3)??g(4)??g(5)??$



- For $n = 3, 4, 5,\ g(n) = 1$
- Is there a general formula in terms of n?

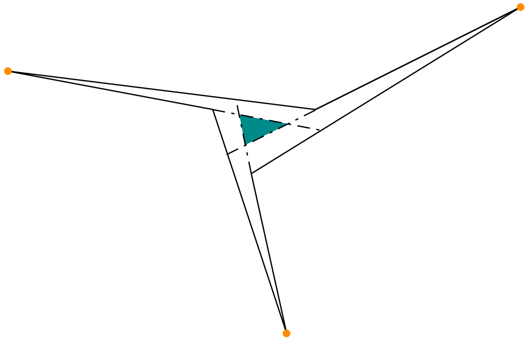
Trying it Out

- Seeing the boundary $\Rightarrow$ seeing the whole interior??

- Even putting guards at every other vertex is not sufficient

Trying it Out

- Seeing the boundary $\Rightarrow$ seeing the whole interior??



- Even putting guards at every other vertex is not sufficient

Art Gallery Theorem

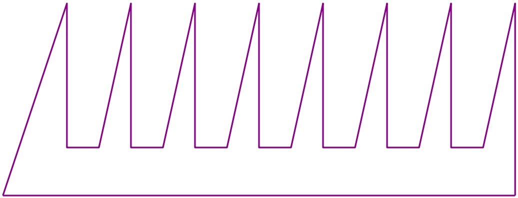
Art Gallery Theorem $g(n) = \lfloor n/3 \rfloor$

- Every n-gon can be guarded with $\lfloor n/3 \rfloor$ vertex guards.
- Some n-gons require at least $\lfloor n/3 \rfloor$ (arbitrary) guards.

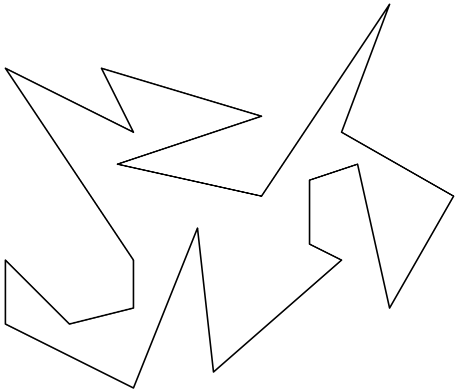
Art Gallery Theorem

Art Gallery Theorem $g(n) = \lfloor n/3 \rfloor$

- Every n-gon can be guarded with $\lfloor n/3 \rfloor$ vertex guards.
- Some n-gons require at least $\lfloor n/3 \rfloor$ (arbitrary) guards.

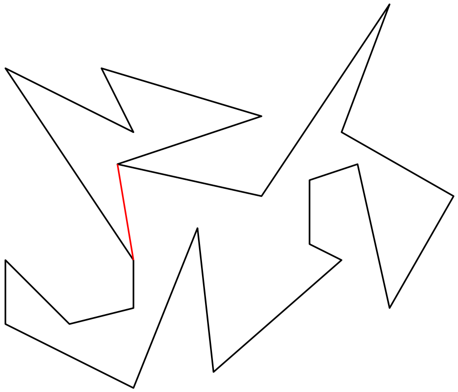


Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



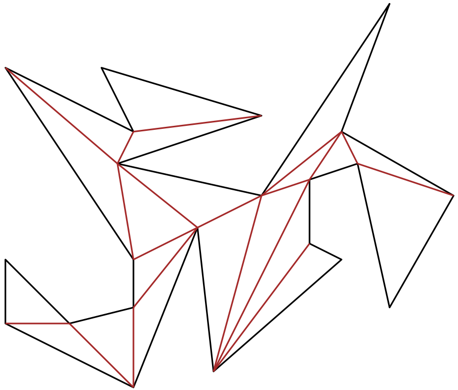
- Diagonal: Given a simple polygon, P, a diagonal is a line segment between two non-adjacent vertices that lies entirely within the interior of the polygon.

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



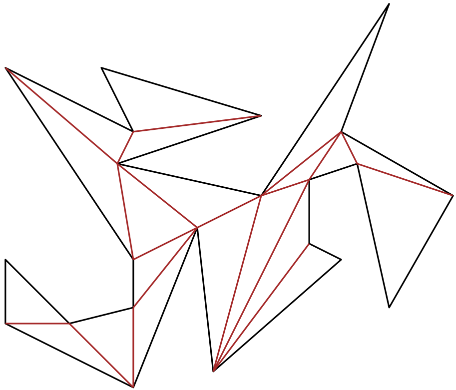
- Diagonal: Given a simple polygon, P, a diagonal is a line segment between two non-adjacent vertices that lies entirely within the interior of the polygon.

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



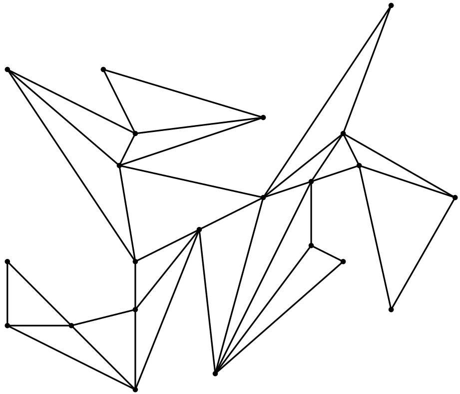
- Triangulations: Given a simple polygon P, a triangulation of P is a partition of the interior of P into triangles using diagonals.

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



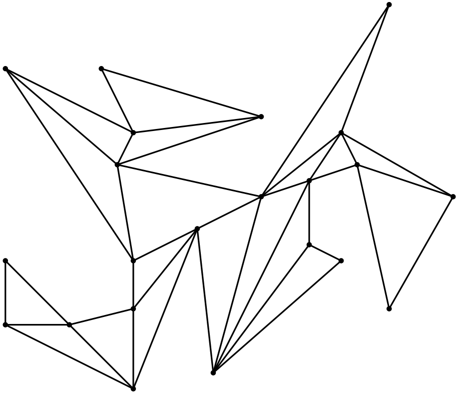
- Observe the polygon P along with the triangulation $\mathcal{T}$ can be considered as a graph $G(P, \mathcal{T})$.
- Vertices: Polygon vertices
- Edges of the graph: Polygon edges $\bigcup$ diagonals of the triangulation

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



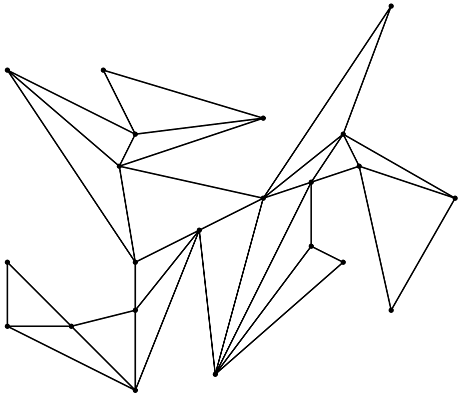
- Observe the polygon P along with the triangulation $\mathcal{T}$ can be considered as a graph $G(P, \mathcal{T})$.
- Vertices: Polygon vertices
- Edges of the graph: Polygon edges $\bigcup$ diagonals of the triangulation

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



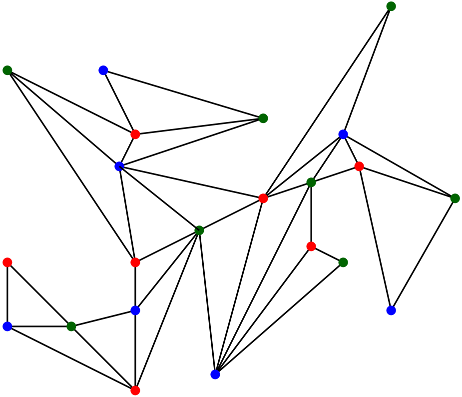
- Properties of the graph
- Planar $\Rightarrow$ Four colorable
- Is it three colorable?

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



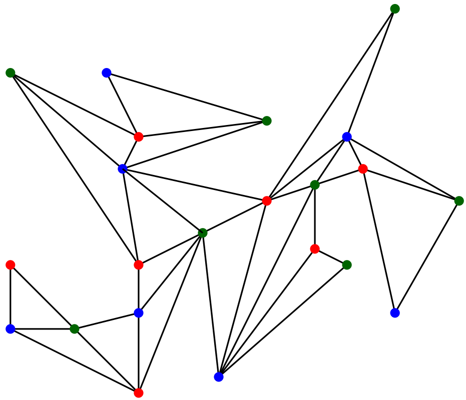
- Properties of the graph
- Planar $\Rightarrow$ Four colorable
- Is it three colorable?

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



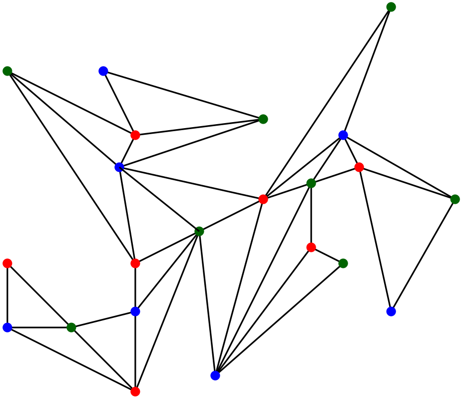
- What if the graph is three colorable
- Does $\lfloor n/3 \rfloor$ guards suffice??

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



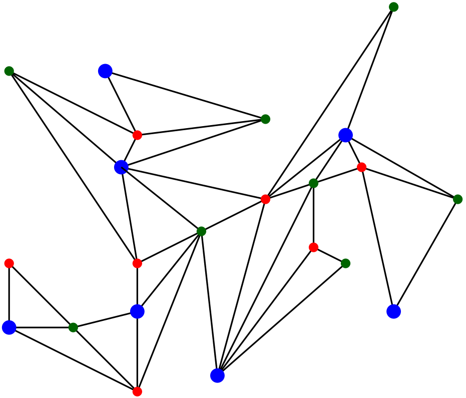
- What if the graph is three colorable
- Does $\lfloor n/3 \rfloor$ guards suffice??

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



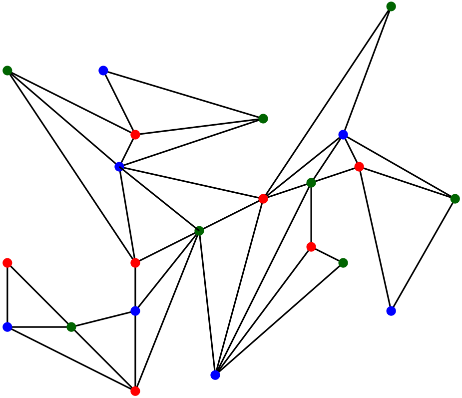
- There exist a color that is used at most $\lfloor n/3 \rfloor$ times
- Post guards at the least popular color vertices

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



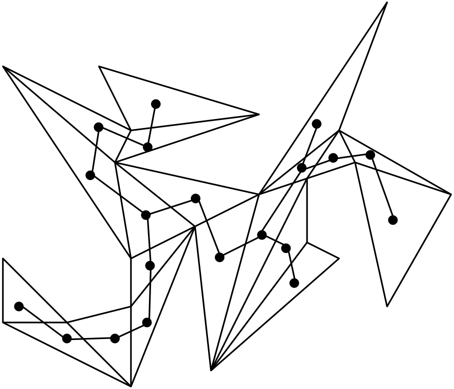
- There exist a color that is used at most $\lfloor n/3 \rfloor$ times
- Post guards at the least popular color vertices

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



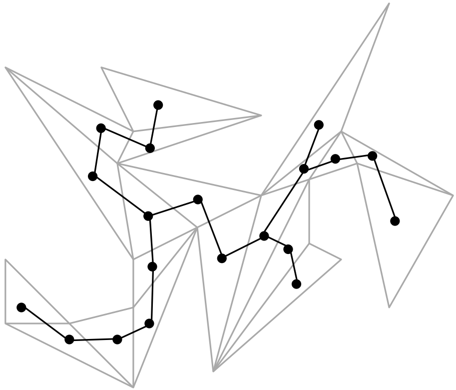
- Why $G(P, \mathcal{T})$ is three colorable?

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



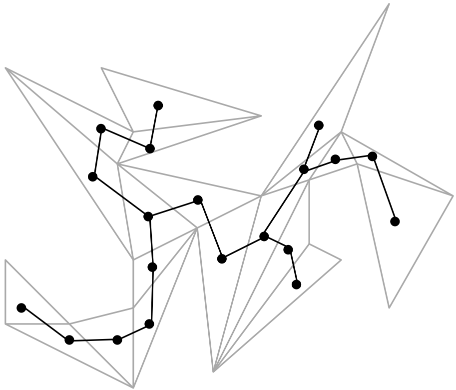
- Dual graph of a polygon: Given a polygon P and a triangulation $\mathcal{T}$ for that polygon, the dual graph is defined as $D(T) = (V, E)$, where $vi \in V$ corresponds to a specific triangle in T, and $(v_a, v_b) \in E$ if the two corresponding triangles share an edge.

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



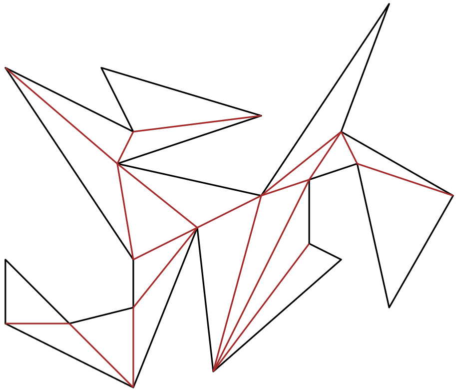
- Dual graph of a polygon: Given a polygon P and a triangulation $\mathcal{T}$ for that polygon, the dual graph is defined as $D(T) = (V, E)$, where $vi \in V$ corresponds to a specific triangle in T, and $(v_a, v_b) \in E$ if the two corresponding triangles share an edge.

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



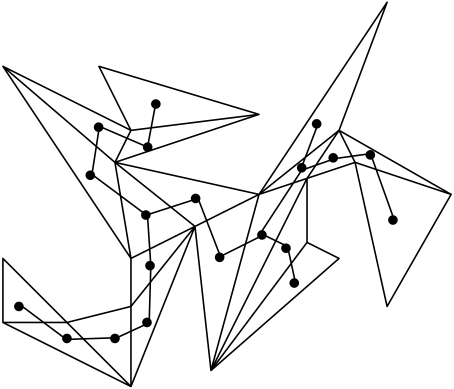
- Lemma: Dual graph of a triangulation of a simple polygon is a tree with maximum degree three.
- Edge of the dual graph corresponds to a diagonal.
- Each diagonal breaks the polygon into two disjoint pieces.

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



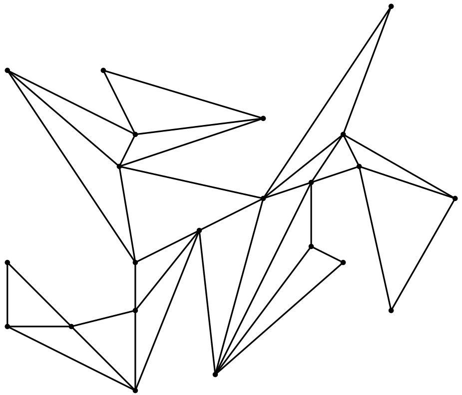
- Lemma: Dual graph of a triangulation of a simple polygon is a tree with maximum degree three.
- Edge of the dual graph corresponds to a diagonal.
- Each diagonal breaks the polygon into two disjoint pieces.

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



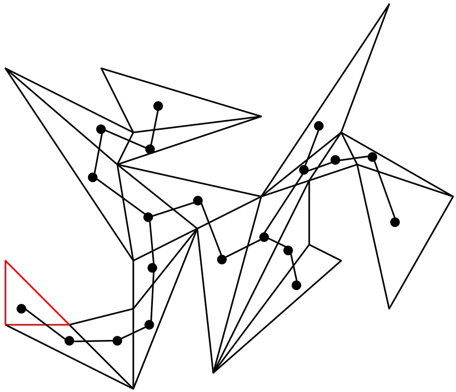
- Lemma: Dual graph of a triangulation of a simple polygon is a tree with maximum degree three.
- Deleting an edge from the dual graph breaks the graph into two connected components.
- Thus the graph is a tree.

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



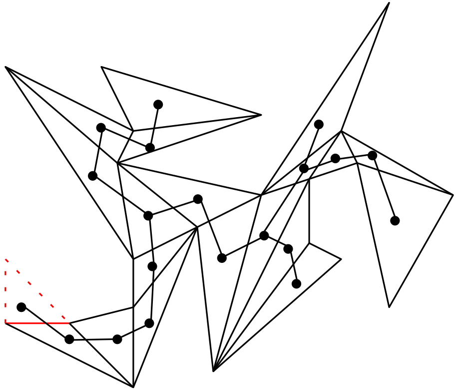
- Lemma: $G(P, \mathcal{T})$ is three colorable
- Proof by Induction:
- Remove a triangle which is a leaf node in the tree.

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



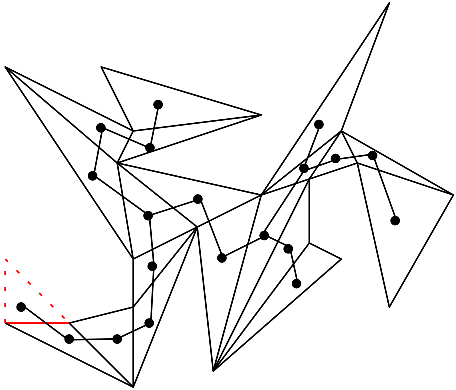
- Lemma: $G(P, \mathcal{T})$ is three colorable
- Proof by Induction:
- Remove a triangle which is a leaf node in the tree.

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



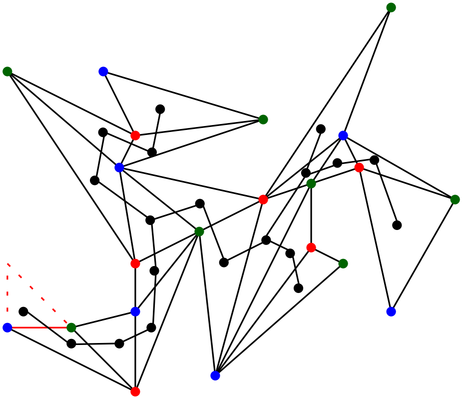
- Lemma: $G(P, \mathcal{T})$ is three colorable
- Proof by Induction:
- Remove a triangle which is a leaf node in the tree.

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



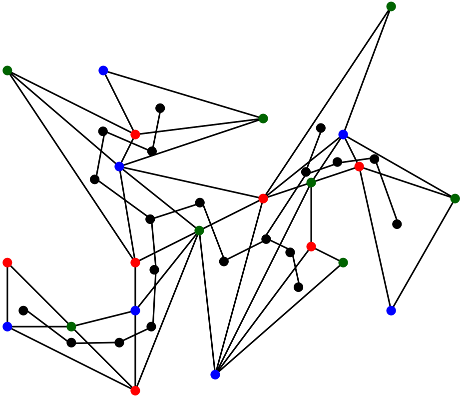
- Inductively 3-color the rest.
- Put the triangle back, coloring new vertex with the label not used by the boundary diagonal.

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



- Inductively 3-color the rest.
- Put the triangle back, coloring new vertex with the label not used by the boundary diagonal.

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice



- Inductively 3-color the rest.
- Put the triangle back, coloring new vertex with the label not used by the boundary diagonal.

Fisk's proof from THE BOOK that $\lfloor n/3 \rfloor$ guards suffice

Theorem

$\frac{n}{3}$ guards are always sufficient and sometimes necessary to guard a simple polygon with n vertices.

Text Book

- Berg, M., D., Kreveld, M., V., Overmars, M., and Schwarzkopf, O., Computational Geometry: Algorithms and Applications, 3rd Edition, Springer, 2008
- Preparata, F., and Shamos, M., Computational Geometry, Springer-Verlag, 1985