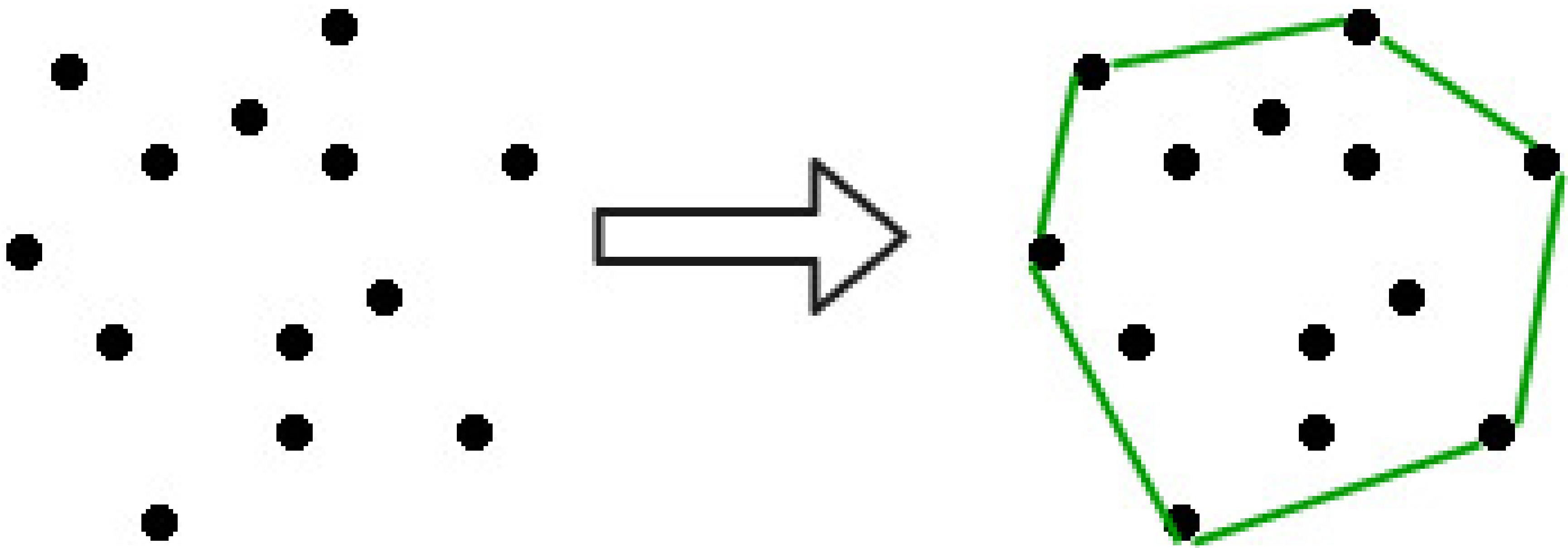
CONVEX HULL

Convex Hull (CH)



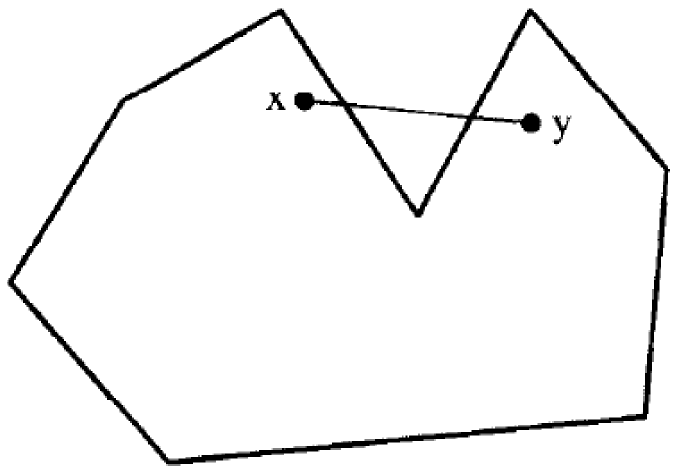
- Closed region including / enclosing all the points

What is special about convex hull?

- A convex hull of a set of points S in the plane is the enclosing convex polygon with:
- Smallest area
- Smallest perimeter

Convex Hull

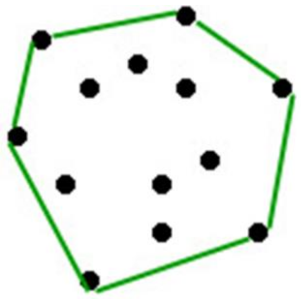
- All the angles are convex



- Any dent implies non-convexity

Standard algorithms for constructing a Convex Hull

Algorithms for CH



- There are many algorithms : least efficient (naïve) to more efficient
- **There can be four types of outputs :**
- All the points on the hull in arbitrary order
- The extreme points (vertices of the convex hull whose interior angle is strictly convex)
- All the points on the hull in boundary traversal order
- The extreme points in boundary traversal order
- Different applications require different outputs

Extreme & non-extreme points

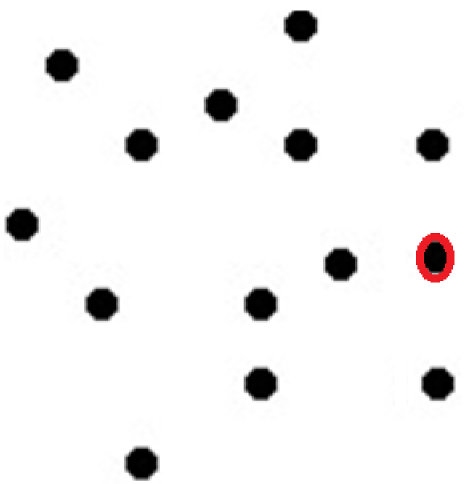
- **Extreme points:**
- The vertices on the boundary of the convex hull whose interior angle is strictly convex
- Examples : The points with largest y coordinate (topmost points)
- Lowest points
- Rightmost points
- Leftmost points
- **Nonextreme points:**
  - Points on the interior of a hull OR
  - points on the boundary of a hull whose interior angle is 180 degree

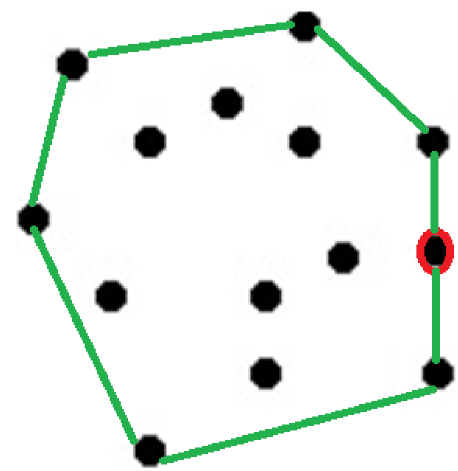
Extreme points

- Exercise: Draw a convex hull with four extreme points and 2 non-extreme points
- Exercise: Draw a convex hull with four extreme points and 2 not strictly convex points on the boundary of the hull
- Actually those points on the boundary of the hull and not strictly convex are not needed for constructing the hull
- Our assumption for hull construction algorithms : More than two points are not collinear

Extreme points

- Recall: those points on the boundary of the hull and not strictly convex are not needed for constructing the hull





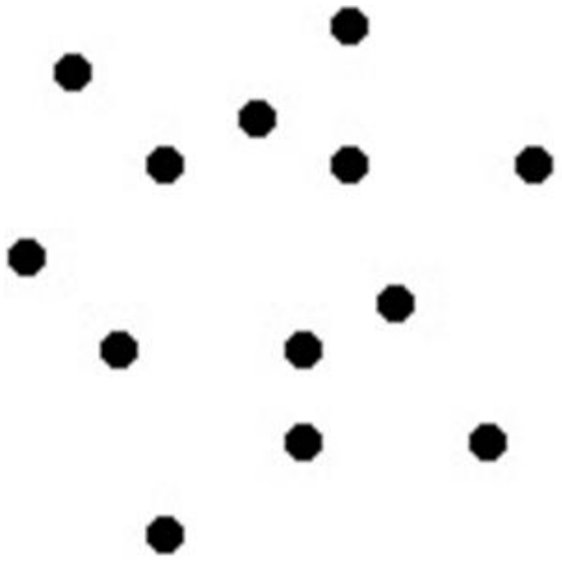
- Recall the assumption for hull algorithms : More than two points are not collinear

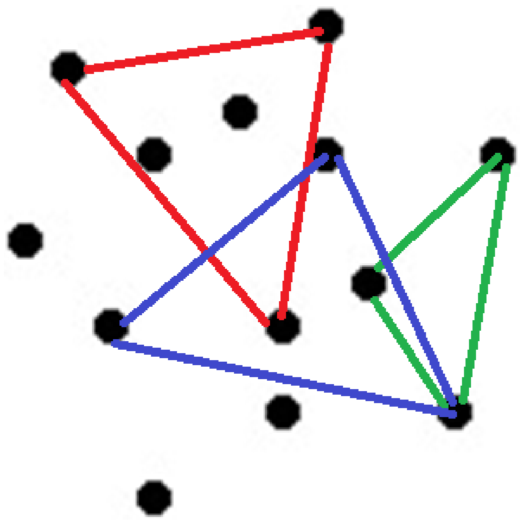
How to find extreme points?

- Rather, how to find non-extreme points

Lemma 3.2.1. *A point is nonextreme iff it is inside some (closed) triangle whose vertices are points of the set and is not itself a corner of that triangle.*

- Exercise: Verify the above lemma empirically





Proof

- If a point is interior to a triangle, it is non-extreme
- Corners of a triangle might be extreme
- A point that lies on the boundary of a triangle but not on a corner is not extreme

Algo : Nonextreme points

Algorithm: INTERIOR POINTS

for each i do

for each $j \neq i$ do

for each $k \neq i \neq j$ do

for each $l \neq i \neq j \neq k$ do

if $p_l \in \triangle(p_i, p_j, p_k)$

then p_l is nonextreme

Complexity

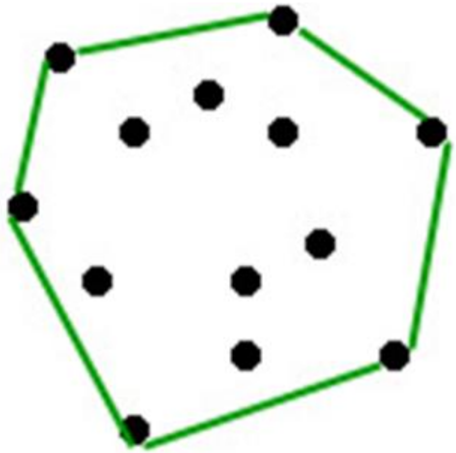
Algorithm: INTERIOR POINTS
for each i do
for each $j \neq i$ do
for each $k \neq i \neq j$ do
    for each $l \neq i \neq j \neq k$ do
        if $p_l \in \Delta(p_i, p_j, p_k)$
            then p_l is nonextreme

- Whether a point is interior to a triangle can be done with three LeftOn operations (Recall reading exercise for LeftOn)
- Complexity of LeftOn is constant
- Complexity of the algo for non-extreme points is ?
- $O(n^4)$

Next algorithm for CH based on extreme edges

Extreme edges

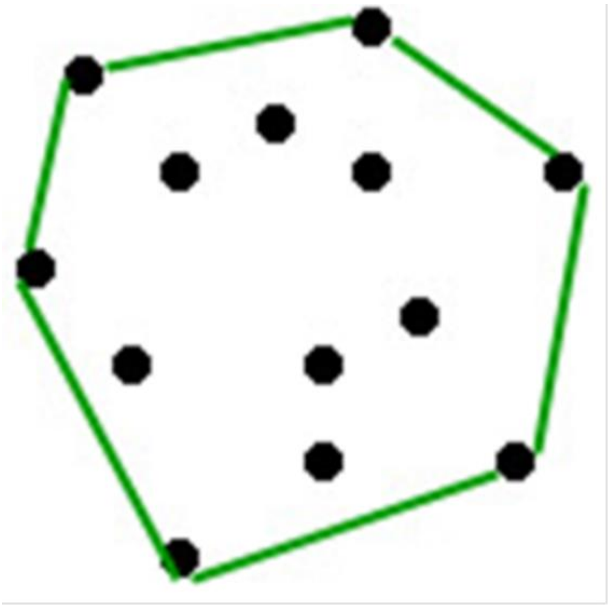
- What are extreme edges?



- Extreme edges are those edges on the convex hull

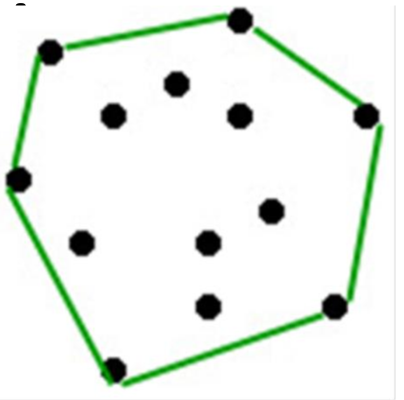
Characteristic of an extreme edge?

- Given a convex hull of S



- An edge is extreme if every point of S are to one side of the line determined by the edge

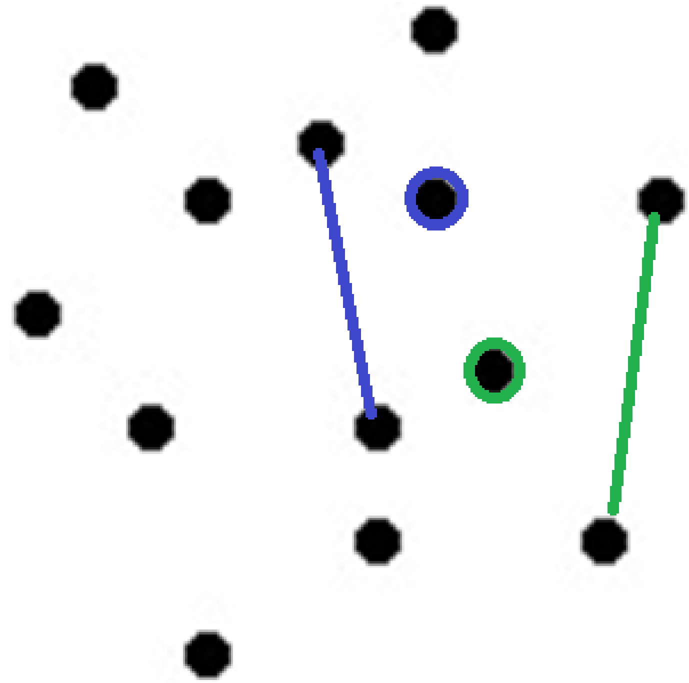
How do we find extreme edges?



- **Assumptions:**
- Treat each edge as a directed edge
- Take one of the two possible directions as determining the side of an edge

- **A directed edge is not extreme if there is some point that is not left of it or on it**

- **A directed edge is not extreme if there is some point that is not left of it or on it**



Algo

Algorithm: EXTREME EDGES

for each i do

for each $j \neq i$ do

for each $k \neq i \neq j$ do

if p_k is *not* left or on (p_i, p_j)

then (p_i, p_j) is not extreme

Time Complexity

- Time Complexity of the algorithm?
- $O(n^3)$

Algorithm: EXTREME EDGES
for each i do
for each $j \neq i$ do
    for each $k \neq i \neq j$ do
        if p_k is *not* left or on (p_i, p_j)
            then (p_i, p_j) is not extreme

More efficient algorithm

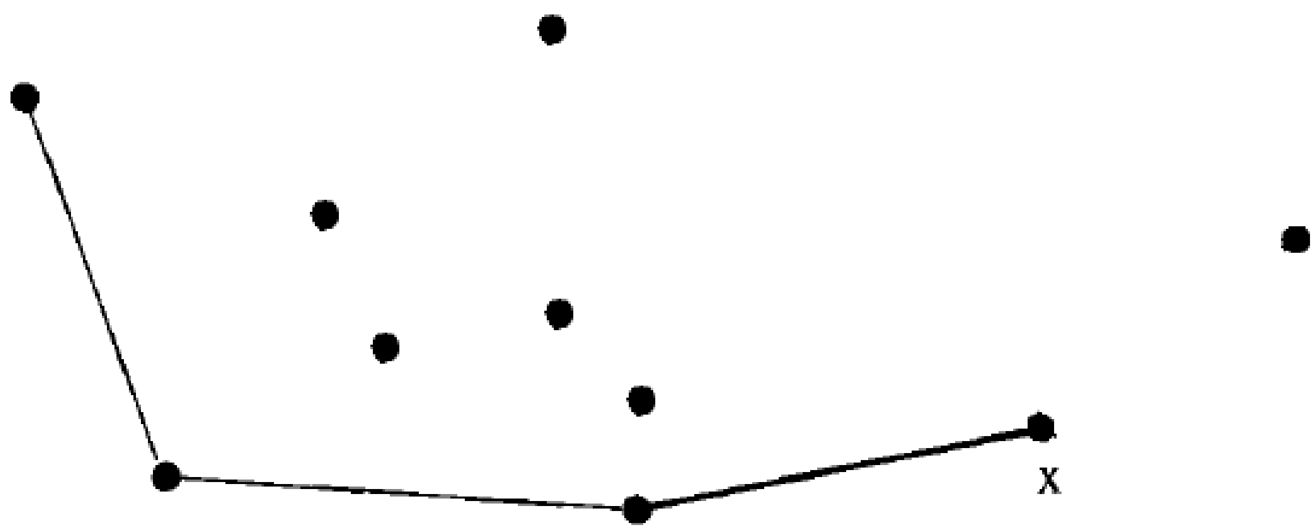
- Name of the algorithm is ***Gift Wrapping***
- What do we learn from the name of the algo?
- We know how do we wrap a gift in general
- We fix one side of the wrapper to the gift, then we wrap the other sides accordingly
- Same concept we apply for constructing Convex hull of a set of points too

Gift Wrapping [Chand & Kapur, 1970]

- Minor variation of the extreme edge algo
- *Gift Wrapping* uses one extreme edge as an anchor for finding the next edge
- **Assumption for the algo**:
- General position of the points OR No three points in S are collinear
- Our aim is to compute an algorithm which takes less than $O(n^3)$ as time complexity

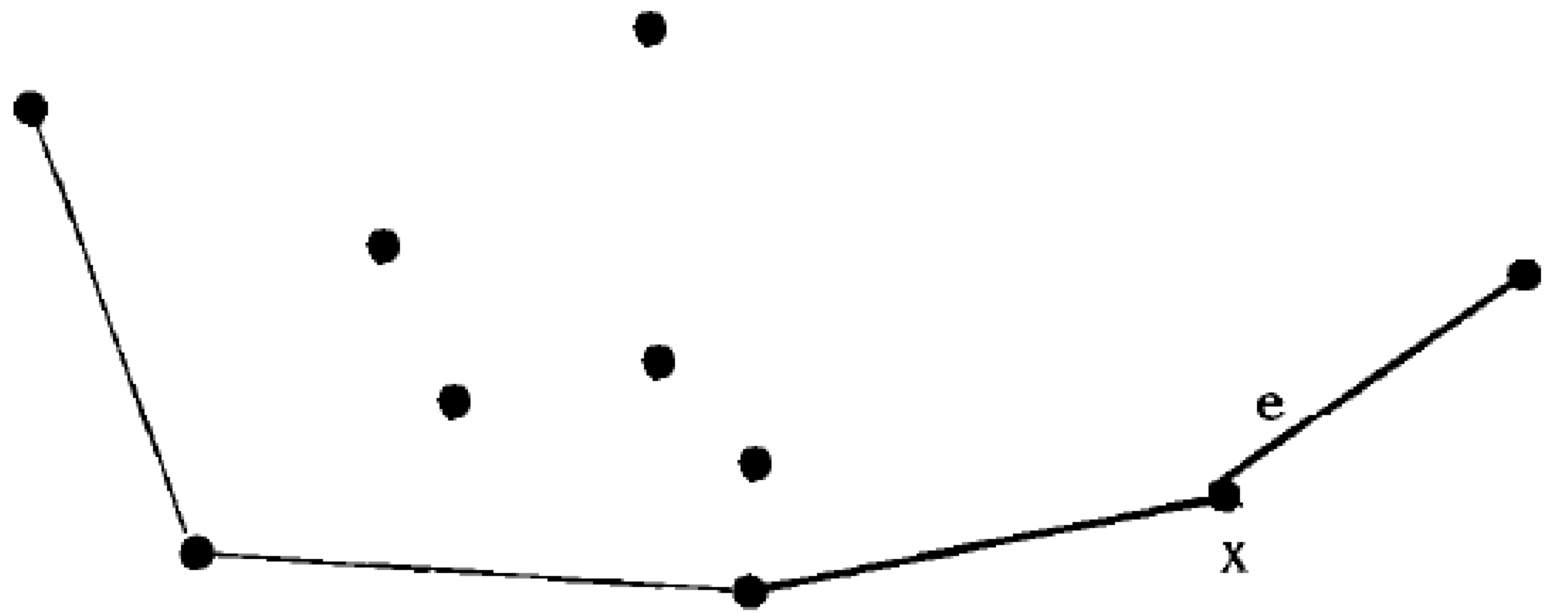
The idea of ***Gift Wrapping***

- Suppose the algorithm last found an extreme edge whose end point is x



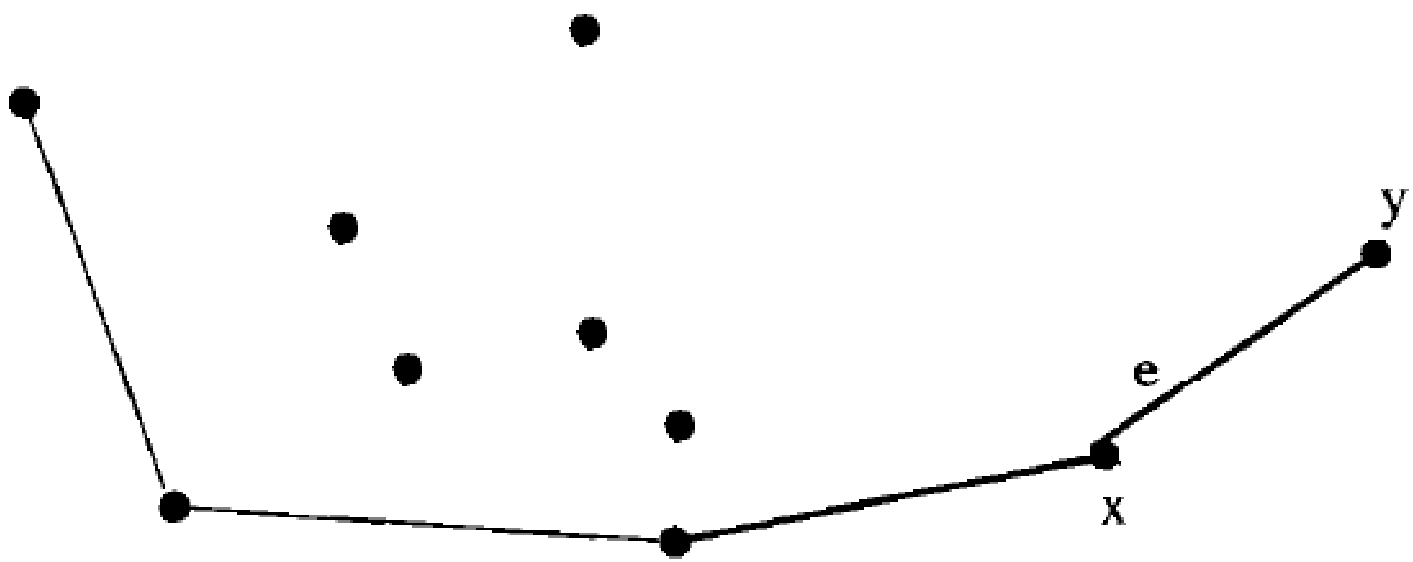
The idea of ***Gift Wrapping***

- We know there must be another extreme edge e sharing endpoint x



The idea of ***Gift Wrapping***

- A directed line L (including the edge e) from x to another point y of S

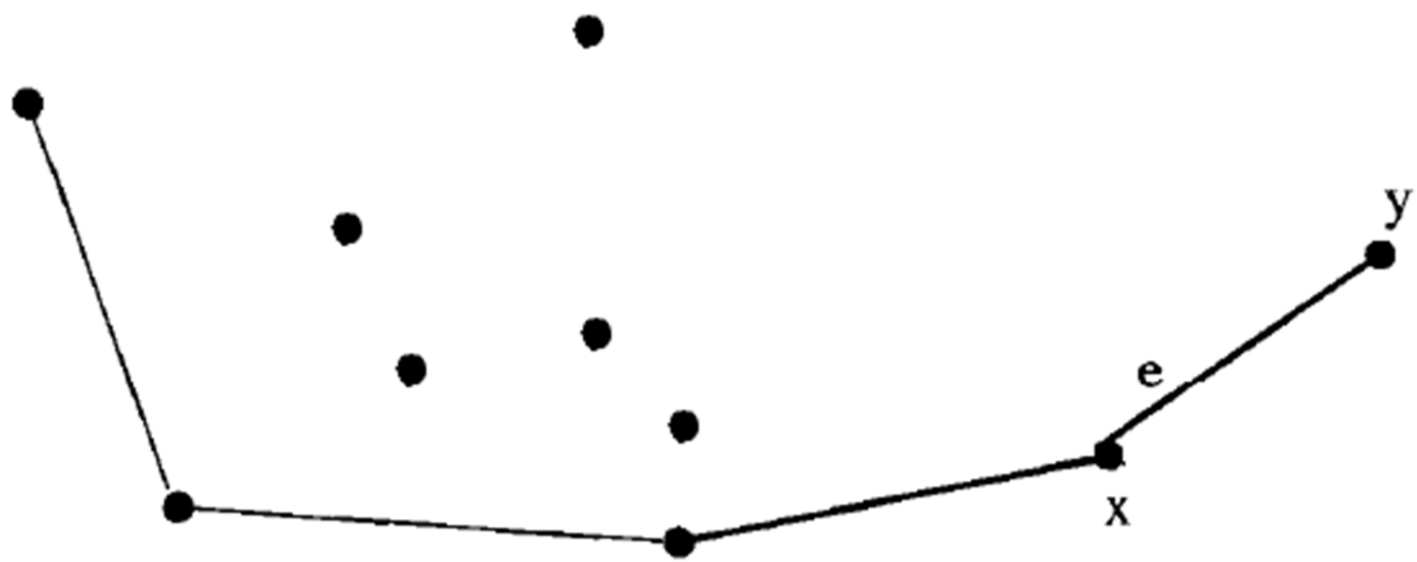


The idea of ***Gift Wrapping***

- The edge e can be included in CH if it is an extreme edge
- What do we have to check for L (the line which includes e) to be an extreme edge?
- All other points of S are to the left of the edge e
- That is for each y, whether all other points are to the left
- For each x, for each y, check all other points, then what will be the time complexity ?
- $O(n^3)$

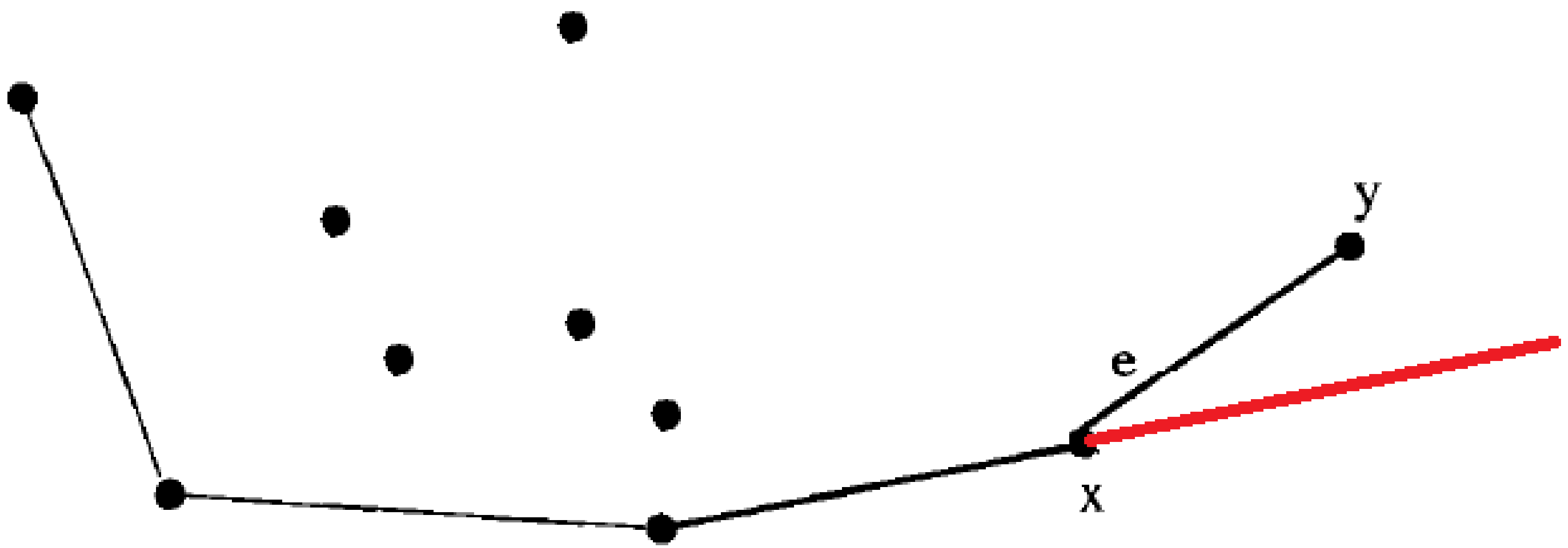
How to reduce the time complexity from $O(n^3)$?

- Exercise: What is the property of the line L with respect to the previous convex hull edge?



How to reduce the time complexity from $O(n^3)$?

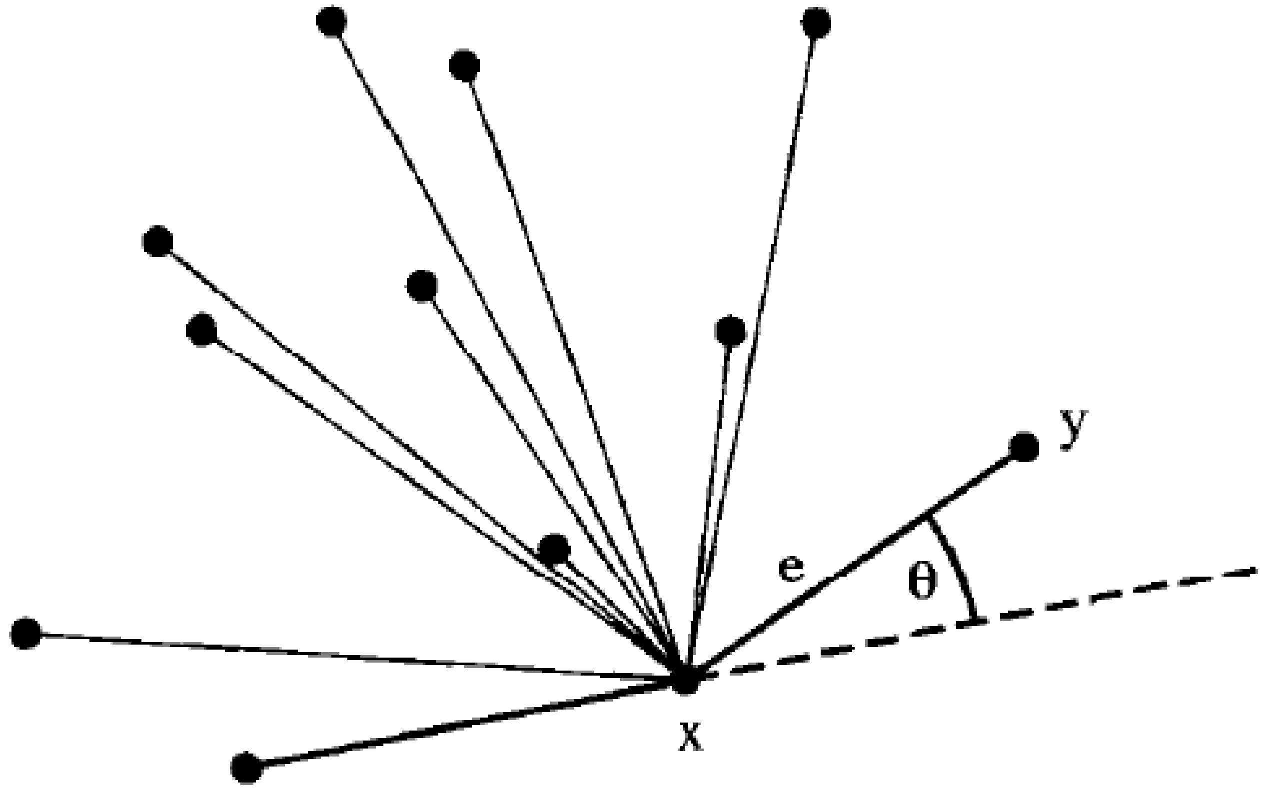
- Is the property more visible when we see the following figure?



The idea of ***Gift Wrapping***

- Select a point y from S, so that xy makes the smallest counter clockwise angle Θ with respect to the previous hull edge which ends in x
- Hence, it is not necessary to check whether all the points are to the left of the new edge e
- It can be inferred from the angle
- Hence, for each point y compute the angle Θ

A general pic

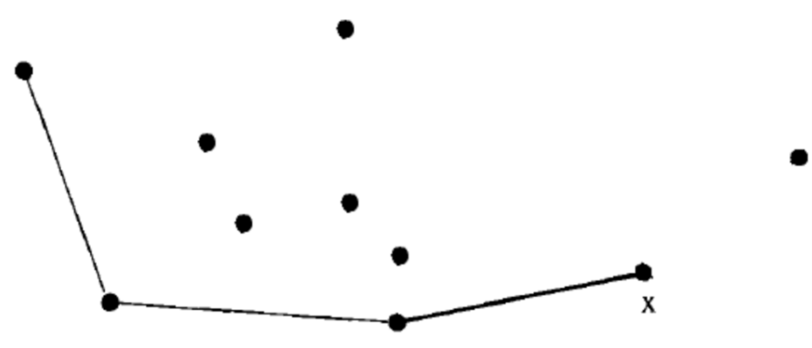


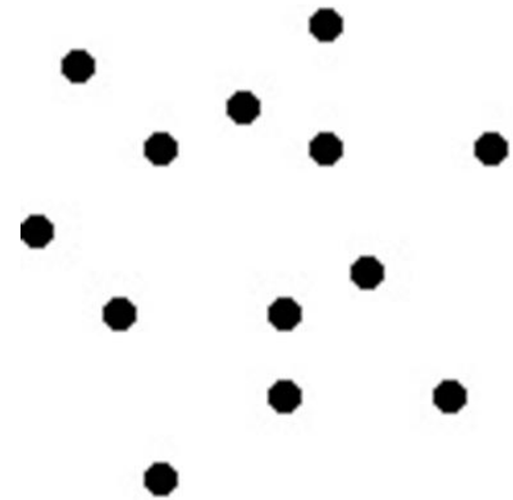
- The point that makes the smallest counter clockwise angle Θ with respect to the previous hull edge must determine an extreme edge

The name : Gift wrapping

- Wrapping the point set with a string that bends the minimum angle from the previous hull edge until the set is hit
- This was first suggested by Chand & Kapur, 1970 as a method for finding hulls in arbitrary dimensions
- **Jarvis March** [R.A. Jarvis, 1973] : Two dimensional case of Gift wrapping
- Exercise : Read the algorithm proposed by Jarvis March

How do we start Gift wrapping?





- **How do we get the first convex hull edge?**
- Find the lowest y coordinate point, i_0
- Draw a line L parallel to x axis along i_0
- Take the counter clockwise angle of the edge i_0y with respect to L for all $y \in S$
- Select the point which takes the smallest counterclockwise angle
- Exercise: Take a set of points and make sure that we understood ***Gift Wrapping***

Pseudo code : Gift Wrapping

Algorithm: GIFT WRAPPING

Find the lowest point (smallest y coordinate).

Let i_0 be its index, and set $i \leftarrow i_0$.

repeat

  for each $j \neq i$ do

    Compute counterclockwise angle θ from previous hull edge.

  Let k be the index of the point with the smallest θ.

  Output (p_i, p_k) as a hull edge.

  $i \leftarrow k$

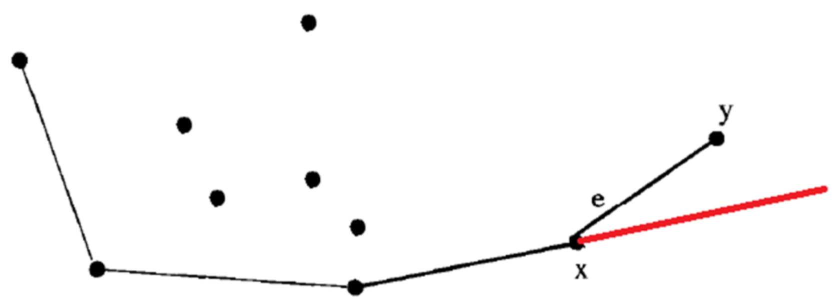
until $i = i_0$

Time Complexity

Algorithm: GIFT WRAPPING

Find the lowest point (smallest y coordinate).
Let i_0 be its index, and set $i \leftarrow i_0$.
repeat
    for each $j \neq i$ do
        Compute counterclockwise angle θ from previous hull edge.
    Let k be the index of the point with the smallest θ.
    Output (p_i, p_k) as a hull edge.
    $i \leftarrow k$
until $i = i_0$

- It depends on what all parameters?
- Number of points n ?
- Number of edges of convex hull h?
- O(nh) is the time complexity
- Hence, Gift wrapping is output sensitive
- It runs faster when the hull is small
- Worst case time complexity ?
- $O(n^2)$: O(n) work for each hull edge



References

- J. O Rourke, *Computational Geometry in C*, 2/e, Cambridge University Press, 1998 *)*

Thank you