Convex Hull Algorithms Overview

convex hull problem and geometric algorithms

The convex hull problem is a fundamental problem in computational geometry that involves finding the smallest convex shape (or "hull") that encloses a given set of points in a 2D or higher-dimensional space. Think of it as wrapping a rubber band around a group of nails on a board; the band takes the shape of the convex hull.

Convex Hull Problem Definition:

Given a set of points $P = \{p_1, p_2, ..., p_n\}$ in a 2D plane, the convex hull is the smallest convex polygon that contains all points in P. Mathematically, a shape is convex if, for every pair of points within the shape, the line segment connecting them also lies entirely within the shape.

Geometric Algorithms for Convex Hull:

Several algorithms can solve the convex hull problem, each with different computational complexities:

1. **Graham's Scan (O(n log n)):**
    - Sorts points by their x-coordinates.
    - Picks a starting point (usually the leftmost point).
    - Uses a stack to determine if a point should be included in the convex hull by checking if moving from one point to another creates a left or right turn.
    - Efficient for moderate-sized datasets.
2. **Jarvis's March (Gift Wrapping) (O(nh)):**
    - Starts from an extreme point and iteratively selects the next point with the smallest angle relative to the previous one.
    - Useful for small datasets but can be slower with larger datasets.
3. **Quickhull (O(n log n) average case):**
    - Similar to QuickSort, it recursively divides the problem into smaller sets.
    - Finds the convex hull by partitioning points into subsets above and below a line segment, solving recursively.
4. **Divide and Conquer (O(n log n)):**
    - Splits the set of points into two halves, computes the convex hull for each half, and merges them.
    - More complex but efficient for large datasets.
5. **Chan's Algorithm (O(n log h)):**

  - A combination of Graham's Scan and Jarvis's March that works well on inputs with a small convex hull.
  - Guarantees O(n log h) complexity, where h is the number of points on the hull.

Applications of Convex Hull:

Convex hulls have many applications, including:

- **Pattern recognition and image processing:** used to identify shapes in image data.
- **Geographical mapping:** helps define boundaries around a set of geographic points.
- **Collision detection in computer graphics:** simplifies the boundary of an object for efficient collision testing.

Understanding these algorithms provides a basis for more advanced geometric algorithms and optimization problems. Would you like examples of implementing any specific algorithm?

what about the brute force approach

The brute force approach to finding the convex hull is straightforward but inefficient, especially for large datasets. It involves checking every possible pair of points to determine if each line segment between them forms an edge of the convex hull. This method has a time complexity of $O(n^3)$, as it requires iterating over all point pairs and confirming if they form part of the convex boundary.

Brute Force Approach Outline:

1. **Iterate through all pairs of points:** For each pair of points (p_i, p_j), treat the line segment connecting them as a potential edge of the convex hull.

2. **Check the line segment direction:** For a line segment between two points (p_i, p_j), determine if all other points in the set lie on one side of this line segment. This can be done using the cross product:

   - Compute the cross product for each point p_k relative to the line formed by (p_i, p_j).
   - If all points are on the same side of the line, then (p_i, p_j) is an edge of the convex hull.

3. **Add the line segment to the hull:** If the line segment passes the above test, add it as part of the convex hull.

Cross Product Check:

To verify that all points lie on the same side of a line defined by points (p_i, p_j), calculate the cross product for each point p_k:

$$\text{cross\_product} = (y_j - y_i) \times (x_k - x_i) - (x_j - x_i) \times (y_k - y_i)$$

- If the cross product is positive for all points, they lie on one side of the line.

- If it’s negative for all points, they lie on the opposite side.

Complexity:

- Since we check each pair of points (p_i, p_j) and iterate over all other points p_k to confirm that each line segment is part of the hull, the time complexity becomes $O(n^3)$.

Example Application:

Though inefficient, the brute force approach can be useful for very small datasets (e.g., less than 10 points) or as a way to verify more efficient algorithms in practice.

Would you like an example of how to code this brute force method?

provide a short numerical problem of that brute force algorithm

Let's walk through a simple numerical example using the brute force algorithm to find the convex hull of a small set of points.

Problem:

Given four points in a 2D plane:

- $P_1 = (0, 0)$
- $P_2 = (1, 1)$
- $P_3 = (2, 0)$
- $P_4 = (1, -1)$

Our goal is to determine which line segments form the convex hull using the brute force approach.

Solution:

1. **Initialize the Points:** We have $n = 4$ points, so we’ll check each pair of points to see if the line segment between them could be an edge of the convex hull.
2. **Examine Each Pair and Use Cross Product:**
    - For each pair of points, check if all remaining points lie on the same side of the line formed by the pair.
3. **Cross Product Calculations:** Use the cross product formula:

$$\text{cross\_product} = (y_j - y_i) \cdot (x_k - x_i) - (x_j - x_i) \cdot (y_k - y_i)$$

   to ensure all other points lie on the same side.

Step-by-Step Solution:

- **Checking** (P_1, P_2)**:**
    - Line formed: $P_1 = (0,0)$ to $P_2 = (1,1)$.
    - Check $P_3 = (2,0)$:

$$\text{cross\_product} = (1-0)\cdot(2-0) - (1-0)\cdot(0-0) = 2$$

      (positive)
    - Check $P_4 = (1,-1)$:

$$\text{cross\_product} = (1-0)\cdot(1-0) - (1-0)\cdot(-1-0) = 2$$

      (positive)

  Since all cross products are positive, (P_1, P_2) is on the hull.

- **Checking** (P_1, P_3)**:**
    - Line formed: $P_1 = (0,0)$ to $P_3 = (2,0)$.
    - Check $P_2 = (1,1)$:

$$\text{cross\_product} = (0-0)\cdot(1-0) - (2-0)\cdot(1-0) = -2$$

      (negative)
    - Check $P_4 = (1,-1)$:

$$\text{cross\_product} = (0-0)\cdot(1-0) - (2-0)\cdot(-1-0) = 2$$

      (positive)

  Points lie on different sides, so (P_1, P_3) is **not** on the hull.

- **Continue Similar Checks for Other Pairs.**

After evaluating all pairs, we find that **the edges of the convex hull** are $(P_1, P_2), (P_2, P_3), (P_3, P_4)$, and (P_4, P_1). These points form the vertices of the convex hull.

Continue Similar Checks for Other Pairs ,

Let's continue with the other pairs and confirm which form the convex hull.

Summary of Points:

- $P_1 = (0,0)$
- $P_2 = (1,1)$

- $P_3 = (2, 0)$
- $P_4 = (1, -1)$

Pairs to Check:

1. Checking (P_1, P_3)

- **Line formed**: $P_1 = (0, 0)$ to $P_3 = (2, 0)$
- **Cross products** with other points:
  - For $P_2 = (1, 1)$:

    $$\text{cross\_product} = (0 - 0) \cdot (1 - 0) - (2 - 0) \cdot (1 - 0) = -2$$

    (negative)
  - For $P_4 = (1, -1)$:

    $$\text{cross\_product} = (0 - 0) \cdot (1 - 0) - (2 - 0) \cdot (-1 - 0) = 2$$

    (positive)
- **Conclusion**: Since P_2 and P_4 lie on opposite sides, (P_1, P_3) **is not on the hull**.

2. Checking (P_1, P_4)

- **Line formed**: $P_1 = (0, 0)$ to $P_4 = (1, -1)$
- **Cross products** with other points:
  - For $P_2 = (1, 1)$:

    $$\text{cross\_product} = (-1 - 0) \cdot (1 - 0) - (1 - 0) \cdot (1 - 0) = -2$$

    (negative)
  - For $P_3 = (2, 0)$:

    $$\text{cross\_product} = (-1 - 0) \cdot (2 - 0) - (1 - 0) \cdot (0 - 0) = -2$$

    (negative)
- **Conclusion**: All cross products are negative, so (P_1, P_4) **is on the hull**.

3. Checking (P_2, P_3)

- **Line formed**: $P_2 = (1, 1)$ to $P_3 = (2, 0)$
- **Cross products** with other points:
  - For $P_1 = (0, 0)$:

    $$\text{cross\_product} = (0 - 1) \cdot (0 - 1) - (2 - 1) \cdot (0 - 1) = 2$$

    (positive)

    - For $P_4 = (1, -1)$:

$$\text{cross\_product} = (0-1) \cdot (1-1) - (2-1) \cdot (-1-1) = 2$$

      (positive)

- **Conclusion**: All cross products are positive, so (P_2, P_3) **is on the hull**.

4. Checking (P_2, P_4)

- **Line formed**: $P_2 = (1, 1)$ to $P_4 = (1, -1)$
- **Cross products** with other points:
    - For $P_1 = (0, 0)$:

$$\text{cross\_product} = (-2-0) \cdot (0-1) - (1-1) \cdot (0-1) = -2$$

      (negative)
    - For $P_3 = (2, 0)$:

$$\text{cross\_product} = (-2-0) \cdot (2-1) - (1-1) \cdot (0-1) = -2$$

      (negative)

- **Conclusion**: All cross products are negative, so (P_2, P_4) **is on the hull**.

5. Checking (P_3, P_4)

- **Line formed**: $P_3 = (2, 0)$ to $P_4 = (1, -1)$
- **Cross products** with other points:
    - For $P_1 = (0, 0)$:

$$\text{cross\_product} = (-1-0) \cdot (0-2) - (1-2) \cdot (0-0) = 2$$

      (positive)
    - For $P_2 = (1, 1)$:

$$\text{cross\_product} = (-1-0) \cdot (1-2) - (1-2) \cdot (1-0) = 2$$

      (positive)

- **Conclusion**: All cross products are positive, so (P_3, P_4) **is on the hull**.

Final Convex Hull Edges:

The edges that form the convex hull are:

1. (P_1, P_2)
2. (P_1, P_4)
3. (P_2, P_3)

4. (P_3, P_4)

Thus, the points forming the convex hull are P_1, P_2, P_3, and P_4, connected in that order.

copy this problem and its complete solution to a new tab

I'm unable to create new tabs, but you can easily copy and paste this problem and solution into a separate document or note for your reference. Alternatively, if there's any specific part you'd like summarized or modified, I can help format it for easy copying!

provide a short and complete numerical problem of Graham's Scan (O(n log n)) with solution

Let's go through a short numerical example using **Graham's Scan** algorithm to find the convex hull of a small set of points.

Problem:

Given five points in a 2D plane:

- $P_1 = (0, 0)$
- $P_2 = (1, 2)$
- $P_3 = (2, 1)$
- $P_4 = (2, -1)$
- $P_5 = (1, -2)$

We want to find the convex hull of these points using Graham's Scan.

Solution:

Steps of Graham's Scan:

1. **Choose the Starting Point**:
    - Select the point with the lowest y-coordinate. If there's a tie, choose the one with the lowest x-coordinate.
    - Here, $P_1 = (0, 0)$ has the lowest y-coordinate, so we select it as our starting point.

2. **Sort Points by Polar Angle**:
    - Calculate the polar angle of each point relative to P_1. Points are sorted in increasing order of this angle.

- For simplicity, we'll use the arctangent of the slope $(y - y_1)/(x - x_1)$ to determine the angles.

Point	Polar Angle with $P_1 = (0, 0)$
P_1	0 (Starting Point)
$P_5 = (1, -2)$	-63.4°
$P_4 = (2, -1)$	-26.6°
$P_3 = (2, 1)$	26.6°
$P_2 = (1, 2)$	63.4°

After sorting based on polar angle, the points are in this order: P_1, P_5, P_4, P_3, P_2.

3. **Construct the Convex Hull**:
    - Start with the first two points in the sorted list, P_1 and P_5, and add them to the convex hull stack.
    - For each subsequent point, check whether moving from the previous two points to this point makes a **left turn** (counter-clockwise) or a **right turn** (clockwise).
    - **Determine Turns**:
        - Use the cross product formula to check for left or right turns:

$$\text{cross product} = (x_j - x_i) \cdot (y_k - y_i) - (y_j - y_i) \cdot (x_k - x_i)$$

          If the result is positive, it's a left turn (counter-clockwise); if negative, it's a right turn (clockwise).
    - **Evaluate Points**:
        - **Point P_4 relative to P_1, P_5**:
            - Cross product $= (1 - 0) \cdot (-1 - 0) - (-2 - 0) \cdot (2 - 0) = 4$ (left turn)
            - Keep P_4 in the stack.
        - **Point P_3 relative to P_5, P_4**:
            - Cross product $= (2 - 1) \cdot (1 - (-2)) - (-1 - (-2)) \cdot (2 - 1) = 3$ (left turn)
            - Keep P_3 in the stack.
        - **Point P_2 relative to P_4, P_3**:
            - Cross product $= (2 - 2) \cdot (2 - (-1)) - (1 - (-1)) \cdot (1 - 2) = 4$ (left turn)
            - Keep P_2 in the stack.
4. **Final Convex Hull**:
    - The points in the stack form the convex hull: P_1, P_5, P_4, P_3, and P_2, which wrap around all points in a convex boundary.

So, the convex hull for this set of points using Graham's Scan is: $(0, 0), (1, -2), (2, -1), (2, 1), (1, 2)$.

Provide a new example in which we skip the points

Let's go through a new example of Graham's Scan where some points will be skipped because they are inside the convex hull.

Problem:

Given six points in a 2D plane:

- $P_1 = (0, 0)$
- $P_2 = (1, 3)$
- $P_3 = (3, 3)$
- $P_4 = (4, 0)$
- $P_5 = (2, 1)$
- $P_6 = (2, 2)$

We aim to find the convex hull of these points using Graham's Scan.

Solution:

Steps of Graham's Scan:

1. **Choose the Starting Point**:
    - Select the point with the lowest y-coordinate. If there's a tie, choose the one with the lowest x-coordinate.
    - Here, $P_1 = (0, 0)$ has the lowest y-coordinate, so we select it as our starting point.
2. **Sort Points by Polar Angle**:
    - Calculate the polar angle of each point relative to P_1. Points are sorted in increasing order of this angle.

Point	Coordinates	Polar Angle with $P_1 = (0, 0)$
P_1	(0, 0)	0 (Starting Point)
P_4	(4, 0)	0°
P_5	(2, 1)	26.6°
P_6	(2, 2)	45°
P_3	(3, 3)	45°
P_2	(1, 3)	71.6°

After sorting based on polar angle, the points are in this order: $P_1, P_4, P_5, P_6, P_3, P_2$.

3. **Construct the Convex Hull**:

  - Start with the first two points in the sorted list, P_1 and P_4, and add them to the convex hull stack.
  - For each subsequent point, check whether moving from the previous two points to this point makes a **left turn** (counter-clockwise) or a **right turn** (clockwise).
  - If it's a right turn, we pop the last point from the stack because it would be inside the convex hull.

4. **Evaluate Points**:

  - **Point P_5 relative to P_1, P_4**:
    - Cross product:

$$(4-0)\cdot(1-0)-(0-0)\cdot(2-0)=4$$

(left turn)

    - Keep P_5 in the stack.

  - **Point P_6 relative to P_4, P_5**:

    - Cross product:

$$(2-4)\cdot(2-1)-(1-0)\cdot(2-4)=-2$$

(right turn)

    - Since it’s a right turn, **pop** P_5 from the stack and re-evaluate P_6 with P_1 and P_4.

    - Re-checking P_6 **relative to** P_1, P_4:
      - Cross product:

$$(4-0)\cdot(2-0)-(0-0)\cdot(2-0)=8$$

(left turn)

      - Keep P_6 in the stack.

  - **Point P_3 relative to P_4, P_6**:

    - Cross product:

$$(2-4)\cdot(3-2)-(2-0)\cdot(3-4)=-2$$

(right turn)

    - Pop P_6 from the stack and re-evaluate P_3 with P_1 and P_4.

    - Re-checking P_3 **relative to** P_1, P_4:
      - Cross product:

$$(4-0)\cdot(3-0)-(0-0)\cdot(3-0)=12$$

(left turn)
    - Keep P_3 in the stack.
  - **Point P_2 relative to P_4, P_3:**
    - Cross product:

$$(3-4)\cdot(3-3)-(3-0)\cdot(3-4)=4$$

(left turn)
    - Keep P_2 in the stack.

5. **Final Convex Hull**:
   - The points remaining in the stack form the convex hull: P_1, P_4, P_3, and P_2.

So, the convex hull for this set of points using Graham's Scan is: $(0,0), (4,0), (3,3), (1,3)$, skipping $P_5 = (2,1)$ and $P_6 = (2,2)$ as they lie inside the convex hull.

why and how Graham's Scan (O(n log n))

Graham's Scan is an efficient algorithm to find the convex hull of a set of points in the plane. It has a time complexity of $O(n \log n)$, primarily because of the sorting step. Here's a breakdown of the "why" and "how" of Graham's Scan, along with the reasons behind its $O(n \log n)$ complexity.

Why Graham's Scan Works

The goal of Graham's Scan is to construct the convex hull by:

1. Selecting the lowest starting point (by y-coordinate).
2. Sorting the remaining points based on their polar angle with respect to the starting point.
3. Using a stack-based approach to ensure each step only adds points that form a "left turn," ensuring a convex shape without inner points.

The steps ensure that:

- **Sorting** helps us easily process points in a way that builds the outer boundary without overlaps.
- **Stack management** (pushing and popping points) ensures that any internal points, which would otherwise create non-convex turns, are removed, and only the outermost points are part of the final convex hull.

How Graham's Scan Works (Detailed Steps)

1. **Find the Starting Point**: Select the point with the lowest y-coordinate. If there is a tie, pick the point with the lowest x-coordinate. This point is guaranteed to be part of the convex hull.

2. **Sort Points by Polar Angle**: Calculate the polar angle between the starting point and each other point, then sort these points in increasing order of this angle. Sorting by angle ensures that we can process points from left to right around the hull.

3. **Scan and Construct the Convex Hull**:
    - Initialize a stack with the starting point and the point with the smallest polar angle.
    - For each subsequent point, check whether adding it would make a "left turn" with the two most recent points in the stack.
        - **Left turn** (counter-clockwise) means the point should be included in the convex hull.
        - **Right turn** (clockwise) means the point makes the shape non-convex, so we pop the last point from the stack and re-evaluate.
    - Continue until all points have been processed, leaving only the points that form the convex hull.

Why the Complexity is $O(n \log n)$

- **Sorting**: Sorting the points by polar angle requires $O(n \log n)$ time, which is the dominant part of the algorithm’s complexity.
- **Stack Operations**: After sorting, each point is pushed and popped from the stack at most once. This contributes $O(n)$ time, which is negligible compared to the $O(n \log n)$ time from sorting.

Therefore, the overall time complexity is $O(n \log n)$.

Summary

- **Sorting Step** $\Rightarrow O(n \log n)$
- **Stack Operations (left/right turn checks)** $\Rightarrow O(n)$

The total time complexity is $O(n \log n)$, making Graham's Scan an efficient solution for finding the convex hull in 2D space, especially for large datasets where $n \log n$ sorting operations dominate over linear operations.

Explain the Jarvis March Algorithm In Easy Way

Sure! The **Jarvis March algorithm**, also known as the **Gift Wrapping algorithm**, is a straightforward method to find the convex hull of a set of points in a 2D plane. Here’s a simple explanation of how it works:

Overview of the Jarvis March Algorithm

1. **Concept**: Imagine you are wrapping a gift. You start at one point and “wrap” around the outermost points until you come back to the starting point. The convex hull is like the outer layer of the gift.

2. **Finding the Convex Hull**: The algorithm works by continuously finding the next outermost point until it returns to the starting point.

Steps of the Jarvis March Algorithm

1. **Start at the Leftmost Point**:
    - Identify the point with the smallest x-coordinate (if there's a tie, choose the one with the smallest y-coordinate). This is your starting point, and it will definitely be part of the convex hull.
2. **Find the Next Point**:
    - From the current point, look at all other points and determine which one makes the smallest angle (or a left turn) with the line formed from the current point.
    - This is done by checking the orientation of every other point relative to the current point. You can use the cross product for this purpose.
3. **Repeat**:
    - Once you find the next point, make it the current point and repeat the process until you return to the starting point.
    - Keep track of all the points that are part of the convex hull.
4. **Finish**:
    - When you return to the starting point, you have completed the convex hull.

Example

Let's say we have the following points:

- $A(1, 1)$
- $B(2, 3)$
- $C(3, 2)$
- $D(4, 1)$
- $E(2, 1)$

Steps in the Example

1. **Identify the Starting Point**:
    - The leftmost point is $A(1, 1)$.
2. **Find the Next Point**:
    - From A, check angles with all other points:
        - Angle with B: left turn
        - Angle with C: left turn
        - Angle with D: left turn
        - Angle with E: right turn
    - The point that makes the most left turn is $B(2, 3)$.

3. **Continue Finding Points**:
   - Now at B, repeat the process:
     - Angle with C: left turn
     - Angle with D: right turn
     - Angle with A: right turn
     - Angle with E: right turn
   - Next point is $C(3, 2)$.

4. **Repeat Until Returning to Start**:
   - Continue until you loop back to A. The points in the convex hull are A, B, C, and D.

Complexity

- **Time Complexity**: The algorithm has a time complexity of $O(n^2)$ because for each point, you check every other point to find the next one.
- **Space Complexity**: It requires $O(n)$ space to store the points in the convex hull.

Summary

- **Jarvis March** is intuitive and easy to understand, making it a great choice for small datasets.
- The algorithm essentially finds the outermost points by wrapping around them like a gift.

This algorithm is simple to implement and visualize, which is why it's often one of the first algorithms taught in computational geometry.

provide a complete step by step easy solution of javris march numerical problem

Let's walk through a complete numerical example using the **Jarvis March algorithm** to find the convex hull of a set of points.

Problem Statement

Given the following points in a 2D plane:

- $P_1 = (0, 0)$
- $P_2 = (1, 2)$
- $P_3 = (2, 1)$
- $P_4 = (2, 3)$
- $P_5 = (3, 2)$
- $P_6 = (0, 3)$

We want to find the convex hull of these points using the Jarvis March algorithm.

Step-by-Step Solution

Step 1: Identify the Starting Point

- Look for the point with the smallest x-coordinate (and smallest y-coordinate in case of a tie).
- The starting point is $P_1 = (0, 0)$.

Step 2: Initialize the Convex Hull

- Create a list to store the points of the convex hull, starting with P_1.

Convex Hull = $[P_1]$

Step 3: Find the Next Point

From the current point P_1, we need to find the next point that makes a left turn:

1. **Current Point**: $P_1 = (0, 0)$
    - Check against all other points:

Point	Orientation (cross product)
P_1	- (Current point)
$P_2 = (1, 2)$	Left turn: $(1 - 0)(2 - 0) - (0 - 0)(2 - 0) = 2$ (positive)
$P_3 = (2, 1)$	Left turn: $(2 - 0)(1 - 0) - (0 - 0)(1 - 0) = 2$ (positive)
$P_4 = (2, 3)$	Left turn: $(2 - 0)(3 - 0) - (0 - 0)(3 - 0) = 6$ (positive)
$P_5 = (3, 2)$	Left turn: $(3 - 0)(2 - 0) - (0 - 0)(2 - 0) = 6$ (positive)
$P_6 = (0, 3)$	Right turn: $(0 - 0)(3 - 0) - (0 - 0)(3 - 0) = 0$ (zero)

    - The point $P_4 = (2, 3)$ is the farthest left with the most positive angle.

Next Point = P_4

Convex Hull = $[P_1, P_4]$

Step 4: Continue Finding Points

Now, set P_4 as the current point and find the next point:

1. **Current Point**: $P_4 = (2, 3)$
    - Check against all other points:

Point	Orientation (cross product)
P_1	Left turn: $(0 - 2)(0 - 3) - (0 - 3)(0 - 2) = 6$ (positive)
$P_2 = (1, 2)$	Left turn: $(1 - 2)(2 - 3) - (3 - 2)(2 - 2) = 1$ (positive)
$P_3 = (2, 1)$	Left turn: $(2 - 2)(1 - 3) - (3 - 2)(1 - 2) = 2$ (positive)
$P_5 = (3, 2)$	Right turn: $(3 - 2)(2 - 3) - (3 - 2)(2 - 2) = -1$ (negative)
$P_6 = (0, 3)$	Left turn: $(0 - 2)(3 - 3) - (3 - 2)(3 - 2) = 0$ (zero)

    - The next point is $P_2 = (1, 2)$.

Convex Hull = $[P_1, P_4, P_2]$

Step 5: Repeat the Process

Set P_2 as the current point:

1. **Current Point**: $P_2 = (1, 2)$
   - Check against all other points:

Point	Orientation (cross product)
P_1	Right turn: $(0-1)(0-2)-(2-2)(0-1)=2$ (negative)
P_4	Right turn: $(2-1)(3-2)-(2-2)(2-1)=-1$ (negative)
$P_3 = (2, 1)$	Left turn: $(2-1)(1-2)-(2-2)(2-1)=1$ (positive)
$P_5 = (3, 2)$	Left turn: $(3-1)(2-2)-(2-2)(2-1)=0$ (zero)
P_6	Right turn: $(0-1)(3-2)-(2-2)(2-2)=1$ (negative)

   - The next point is $P_3 = (2, 1)$.

Convex Hull = $[P_1, P_4, P_2, P_3]$

Step 6: Continue Until Back to Start

Set P_3 as the current point:

1. **Current Point**: $P_3 = (2, 1)$
   - Check against all other points:

Point	Orientation (cross product)
P_1	Right turn: $(0-2)(0-1)-(1-1)(1-2)=2$ (negative)
P_4	Right turn: $(2-2)(3-1)-(1-2)(1-1)=0$ (zero)
P_2	Left turn: $(1-2)(2-1)-(1-1)(1-2)=1$ (positive)
$P_5 = (3, 2)$	Left turn: $(3-2)(2-1)-(1-1)(1-2)=1$ (positive)
P_6	Right turn: $(0-2)(3-1)-(1-1)(1-2)=2$ (negative)

   - The next point is $P_5 = (3, 2)$.

Convex Hull = $[P_1, P_4, P_2, P_3, P_5]$

Final Step: Complete the Hull

1. **Current Point**: $P_5 = (3, 2)$
   - Check against all other points:

Point	Orientation (cross product)
P_1	Right turn: $(0-3)(0-2)-(2-2)(2-2)=6$ (negative)
P_4	Right turn: $(2-3)(3-2)-(2-2)(2-2)=1$ (negative)
P_2	Right turn: $(1-3)(2-2)-(2-2)(2-1)=0$ (zero)

Point	Orientation (cross product)
P_3	Right turn: $(2-3)(1-2)-(2-2)(2-2)=1$ (negative)
P_6	Right turn: $(0-3)(3-2)-(2-2)(2-2)=3$ (negative)

- All other points cause right turns, so we stop here.

2. **Return to Start**: Now that we've looped back to the starting point P_1, we conclude the process.

Final Convex Hull

The points in the convex hull are:

- $P_1 = (0, 0)$
- $P_2 = (1, 2)$
- $P_4 = (2, 3)$
- $P_5 = (3, 2)$
- $P_3 = (2, 1)$

Summary

The final convex hull formed by the points is:

- $(0, 0), (1, 2), (2, 3), (3, 2), (2, 1)$

The Jarvis March algorithm effectively wraps the points around the outermost boundary, which is the convex hull.

The next point is P2=(1,2) why?

Let's clarify the reasoning behind choosing $P_2 = (1, 2)$ as the next point in the Jarvis March algorithm. We will focus on how we determine the next point based on the orientation of the other points relative to the current point.

Context Recap

1. **Current Point**: We started with $P_4 = (2, 3)$ as the current point in the previous step of the algorithm.

2. **Finding the Next Point**: We need to check the orientation of each point relative to P_4 to determine which point makes the smallest angle or, more generally, which point results in a left turn (counter-clockwise turn) from the current point P_4.

Orientation Calculation