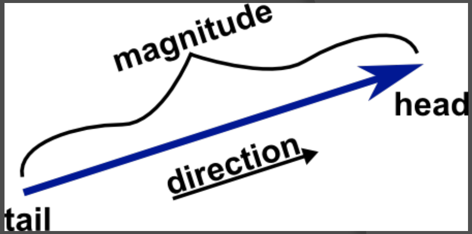
Advanced Computer Contest Preparation
Lecture 19

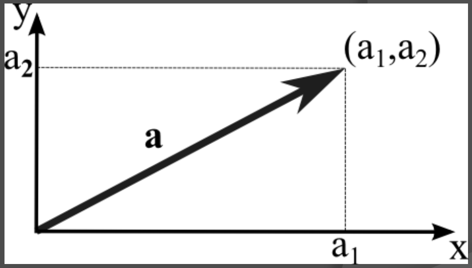
# COMPUTATIONAL GEOMETRY

# Vectors

# Vectors

- Referring to mathematical vectors, not the data structure!
- Two main types
  - Euclidean vector (magnitude, direction)
  - Coordinates
    - Most frequently used
- Representation is typically done using **pair**
  - **pair<obj,obj>**



# `pair` Tips

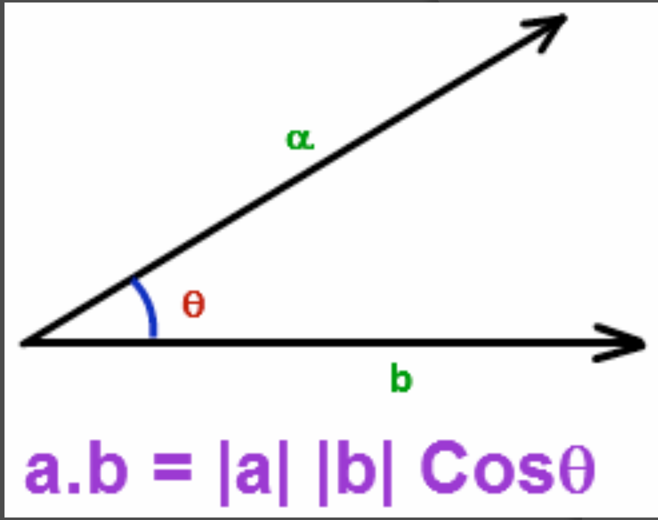
- Can use `typedef` or `#define` to reduce amount of code
- `typedef`:

```
typedef pair<int,int> pii;
pii p;
```

- `#define`:

```
#define pii pair<int,int>
#define F first
pii p;
p.F = 2;
```

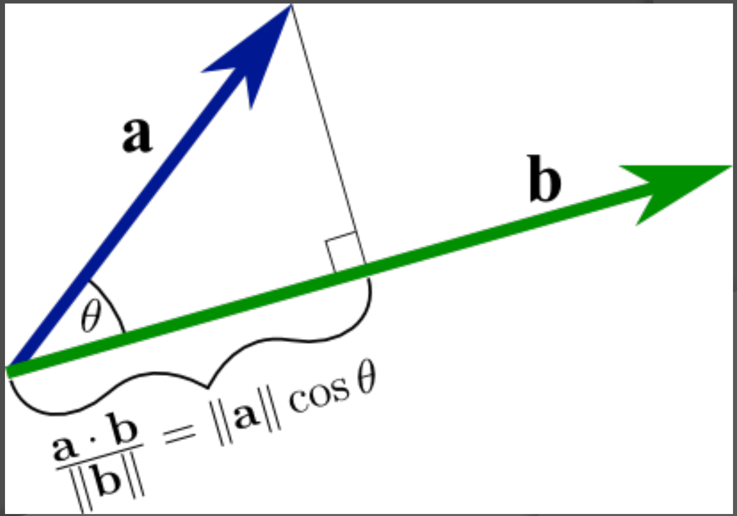
# Dot Product



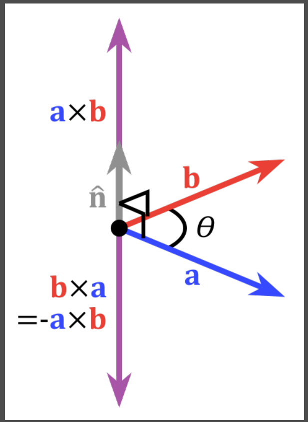

- $\mathbf{a} = [a_1, a_2, \ldots, a_n]$, $\mathbf{b} = [b_1, b_2, \ldots, b_n]$
- Algebraic definition
  - $\mathbf{a} \bullet \mathbf{b} = a_1 b_1 + a_2 b_2 + \ldots + a_n b_n$
- Geometric definition
  - $\mathbf{a} \bullet \mathbf{b} = |\mathbf{a}||\mathbf{b}| \cos\theta$
  - $|\mathbf{v}|$ denotes the magnitude of $\mathbf{v}$
    - The straight line distance from $[0,0,\ldots]$ to $\mathbf{v}$
  - $\theta$ is the angle between $\mathbf{a}$ and $\mathbf{b}$

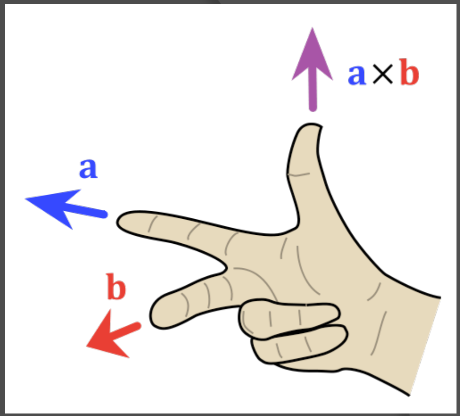
# Dot Product Applications

- $a_1b_1 + a_2b_2 + \ldots + a_nn_n = |\mathbf{a}||\mathbf{b}|\ cos\ \theta$
- Angle
  - Can solve for $\theta$
  - Gives the smallest angle between vectors $\mathbf{a}$ and $\mathbf{b}$
- Scalar Projection
  - $|\mathbf{a}|\ cos\ \theta$
  - "The magnitude in which $\mathbf{a}$ covers $\mathbf{b}$"



# Cross Product

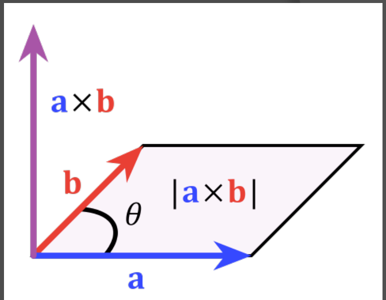



- $\mathbf{a} = [a_1, a_2, a_3]$, $\mathbf{b} = [b_1, b_2, b_3]$
- Geometric definition
  - $\mathbf{a} \times \mathbf{b} = |\mathbf{a}||\mathbf{b}| \sin\theta\, \mathbf{n}$
    - $\mathbf{n}$ is a unit vector (magnitude 1) perpendicular to both $\mathbf{a}$ and $\mathbf{b}$
    - Direction depends on right-hand rule
- Algebraic definition
  - $\mathbf{a} \times \mathbf{b} = [a_2b_3 - a_3b_2, a_3b_1 - a_1b_3, a_1b_2 - a_2b_1]$
  - If $\mathbf{a}$ and $\mathbf{b}$ are vectors in 2-D space, $a_3 = b_3 = 0$ and $\mathbf{a} \times \mathbf{b} = [0, 0, a_1b_2 - a_2b_1]$

# Cross Product Applications

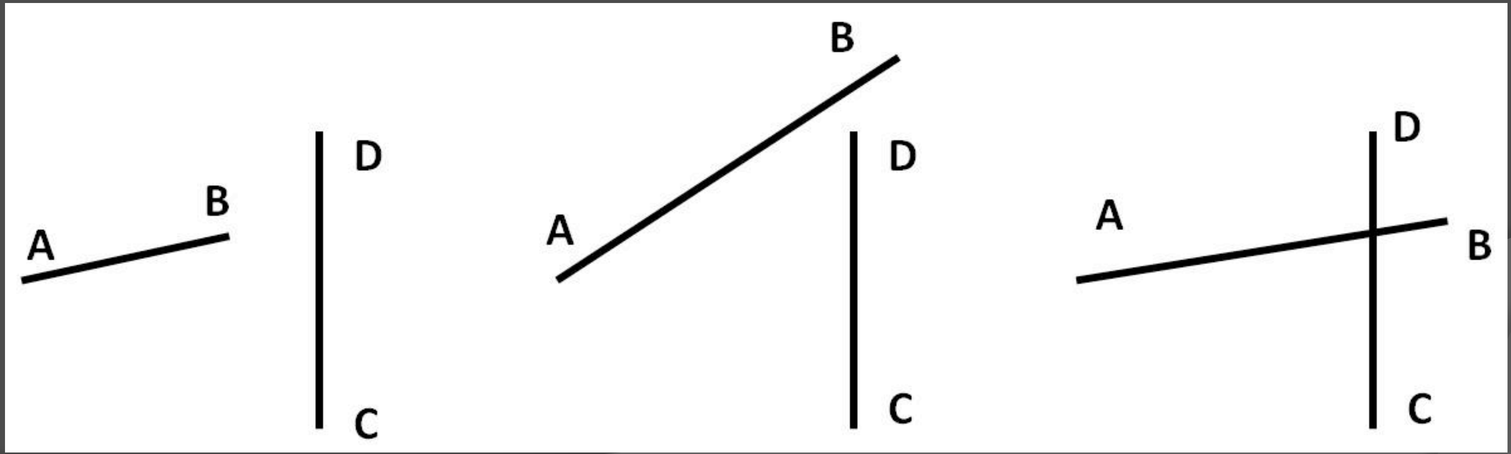
- Area
  - Area of a triangle given 2 sides and their angle is $\frac{1}{2}\, a\, b \sin\theta$ (half of a parallelogram)
  - $a\, b \sin\theta$ can be quickly computed using cross product
- Turn direction
  - If $\mathbf{b}$ is a counter-clockwise turn from $\mathbf{a}$, $\sin\theta$ is positive
  - If $\mathbf{b}$ is a clockwise turn from $\mathbf{a}$, $\sin\theta$ is negative
  - Positive/negative check of cross product can be used to check for clockwise/counter-clockwise turn



# Line Intersection

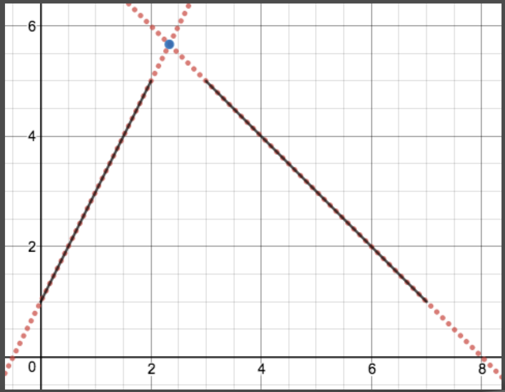
# Problem: Line Intersection

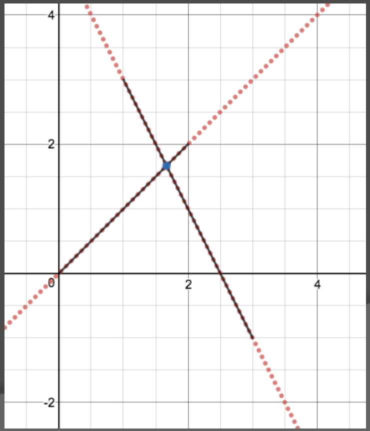
- You are given 2 line segments in 2-D space
- For each line segment, the coordinates of the endpoints are given
- Do the lines intersect?



# Algebraic Approach

- Represent lines as linear equations
- Solve the system of two equations and two unknowns
- Check if the point of intersection is within the endpoints of the two lines



# Algebraic Approach

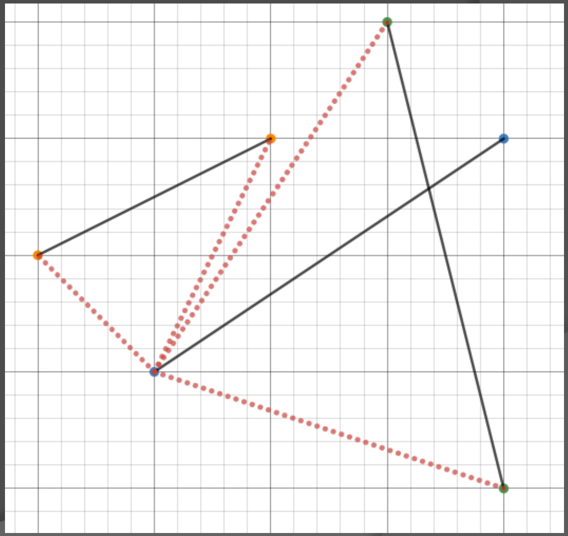
- Slope-Intercept form
  - $y = mx + b$
- Easy to solve
- Must handle case if slope is vertical (undefined)
- Must handle case if lines are parallel

# Algebraic Approach

- Standard Form
  - $Ax + By + C = 0$
- Must find values of $A$, $B$, and $C$
- Can use Cramer's Rule to solve
- Can handle cases where slope is undefined
- Must check if lines are parallel
  - Either no solution or infinite solutions

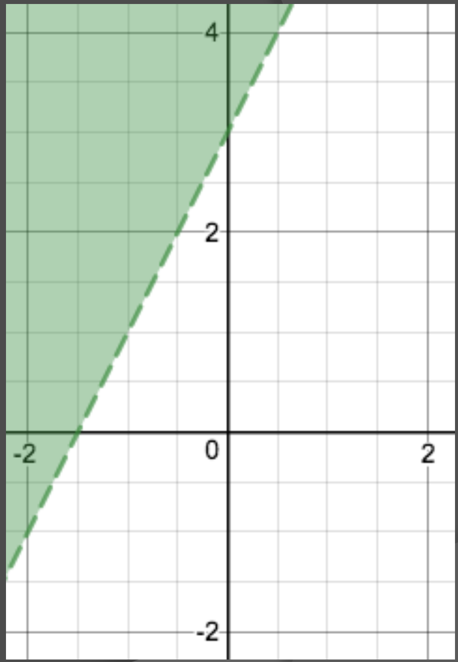
# Vector Approach

- Choose one line
- The other line's endpoints must be on opposite sides of the chosen line
- Use cross product to determine if one endpoint is a clockwise turn from chosen line (vector) and the other is counter-clockwise
- Magnitude of cross product does not matter – only sign matters
  - Be careful not to overflow!
- Repeat with other line to ensure intersection



# Inequality Approach

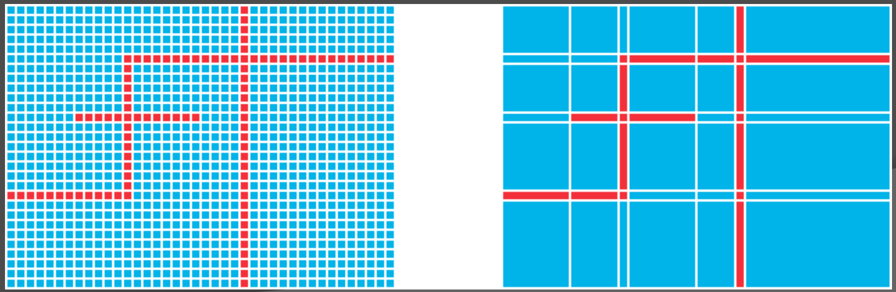
- A linear inequality splits the entire 2-D plane in half
  - $Ax + By + C > 0$
- Find the equation of one of the lines
- In order for an intersection to occur, the other line's endpoints must be on opposite sides
  - One endpoint satisfies the inequality, the other does not
- Repeat with other line to ensure intersection



# Coordinate Techniques

# Coordinate Compression

- Coordinates often span large ranges
  - For example, $(-10^9 <= x,y <= 10^9)$
- Coordinate compression takes the significant $x$ and $y$ values and ignores the rest
  - Takes advantage of large empty spaces
- Take unique coordinate values and sort in increasing order
  - Easiest method is to use a map for each dimension
- Assign an index value to each value



# Coordinate Compression Example

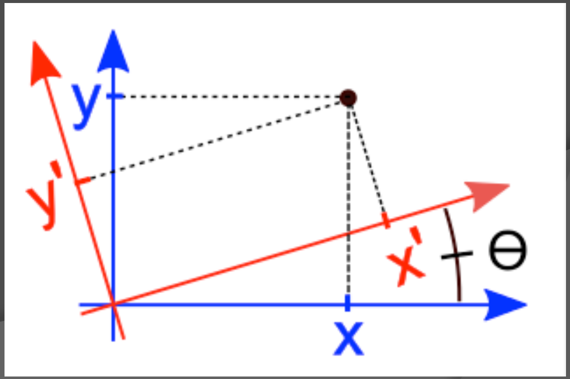
- Given $N$ points, what is the smallest axes-aligned rectangle that contains $K$ points?
- Compress coordinates
- Use a 2-D prefix sum array to keep track of total number of points
- Try every possible rectangle, choose the smallest rectangle with at least $K$ points
- Runtime: $O(N^4)$

# Coordinate Rotation

- Coordinates may need to be rotated to make overall algorithm easier
- Determine equations of lines of new axes in standard form
- Distance to the two new axes are the new $x$ and $y$ coordinates
- Use distance from point to line formula:

$$\frac{Ax + By + C}{\sqrt{A^2 + B^2}}$$

- Remove the square root scale (denominator of formula)



# Coordinate Rotation

- Most common example: squares/rectangles at 45° to $x$ and $y$ axes
  - $y = x$ and $y = -x$ are the new $x$ and $y$ axes, respectively
    - In standard form, $-x + y + 0 = 0$, $x + y + 0 = 0$
  - Distance from point to $y = x$ is new $y$ coordinate
  - Distance from point to $y = -x$ is new $x$ coordinate
  - Use distance from line formula

    $$(x', y') = (\frac{x+y}{\sqrt{2}}, \frac{-x+y}{\sqrt{2}})$$
  - Can remove the $\frac{1}{\sqrt{2}}$ scale

    $$(x', y') = (x + y, y - x)$$

# THANK YOU!