Geometría y Planaridad

Agustín Santiago Gutiérrez

(Basado en diapositivas de Melanie Sclar en TC 2014 Argentina)

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Campamento Caribeño ACM-ICPC 2016

Contenidos

“He who despises Euclidean Geometry is like a man who, returning from foreign parts, disparages his home.”

H. G. Forder.

“Que nadie entre aquí si no sabe Geometría.”

Platón

“Las ecuaciones son sólo la parte aburrida de la matemática. Trato de ver las cosas en términos de geometría”

Stephen Hawking

Generalmente, los problemas de geometría de ACM requieren calcular alguna cantidad (por ej., una distancia, un área, etc) relacionada con elementos geométricos como puntos, líneas, círculos, etc. Para resolverlos, debemos poder ser capaces de:

1. Representar los objetos geométricos involucrados en el problema, para poder operar con ellos.
2. Desarrollar un algoritmo para buscar la respuesta deseada.

Entonces, para poder llegar a la segunda parte (la más interesante), primero tenemos que ser capaces de llevar a cabo la primera.

Nos referiremos principalmente a problemas en $\mathbb{R}^2$ por ser más comunes, pero las mismas ideas aplican a $\mathbb{R}^3$ y superiores.

Como idea general, la técnica más básica para problemas geométricos es la de **discretización de candidatos**:

- Muchas veces se pide encontrar, por ejemplo, un punto del plano que cumpla alguna característica particular.
- Pero como el plano contiene infinitos puntos, no podemos probarlos todos.
- Casi siempre podemos identificar un subconjunto de **candidatos**, de manera que la respuesta buscada sí o sí sea alguno de ellos.
- En este caso, una solución posible consiste en explorar todos los candidatos y verificar lo que corresponda para cada uno.

Es fácil pasar por alto estas ideas, y tratar de buscar una solución más complicada “que encuentre la solución directamente” si uno no está atento.

Contenidos

Un punto en el plano (o un vector desde el origen hasta dicho punto) se puede representar con un par ordenado de coordenadas en un sistema cartesiano:

$$\vec{P} = (x, y) \text{ con } x, y \in \mathbb{R}$$

Se puede representar como `pair<double,double>`, pero es poco declarativo. Mejor hacer un `struct`:

```
1 | struct Punto {
2 |         double x;
3 |         double y;
4 |         // double z; para problemas en el espacio, etc.
5 | };
```

- La suma y resta de vectores se realiza componente a componente:

$$\vec{P} = \vec{P}_1 \pm \vec{P}_2 \qquad \Longleftrightarrow \qquad (x, y) = (x_1 \pm x_2, y_1 \pm y_2)$$

- La longitud o *norma* de un vector es $|\vec{P}| = \sqrt{x^2 + y^2}$.
- La distancia euclídea entre dos puntos $\vec{P}_1$ y $\vec{P}_2$ es $|\vec{P}_1 - \vec{P}_2|$

Producto escalar entre vectores

Producto escalar

El *producto escalar* de dos vectores $u = (x_1, y_1)$ y $v = (x_2, y_2)$ es un número, y se define como

$$\vec{u} \cdot \vec{v} = x_1 x_2 + y_1 y_2 = |\vec{u}||\vec{v}| \cos\theta$$

$$\vec{u} \cdot \vec{v} = x_1 x_2 + y_1 y_2 + z_1 z_2 = |\vec{u}||\vec{v}| \cos\theta \text{ (en 3D)}$$

- El valor del producto escalar es la proyección ortogonal de u sobre v, expresada con signo y multiplicada por $|v|$. Notar que como es conmutativo, lo mismo vale al revés.
- Observar que en particular si $\theta = 90°$ el producto escalar se anula. **Es decir, dos vectores son perpendiculares si y sólo si su producto escalar da 0**.
- **El producto escalar funciona igual, y tiene estas mismas propiedades, en** $\mathbb{R}^n$ (por ejemplo, proyectar en 3D).

Producto cruz entre vectores

El producto vectorial (o producto cruz) de dos vectores es un *vector* en la dirección normal al plano generado por ellos (¡esto puede ser muy útil para problemas en $\mathbb{R}^3$!).
Si $u, v \in \mathbb{R}^2$ ($u = (x_1, y_1)$ y $v = (x_2, y_2)$), se define $u \times v$ como:

$$\vec{u} \times \vec{v} = x_1 y_2 - x_2 y_1$$

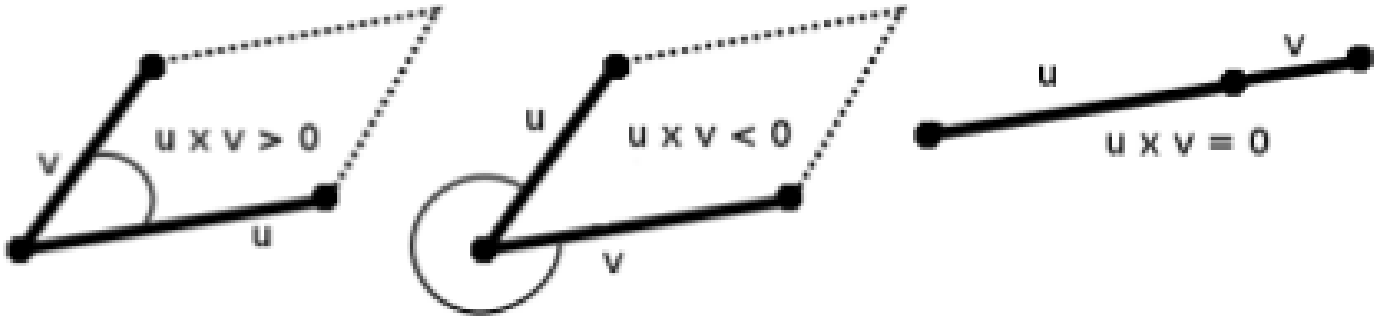
que corresponde a la componente z del producto vectorial, y se puede demostrar que:

$$|\vec{u} \times \vec{v}| = |\vec{u}||\vec{v}| \sin\theta,$$

y que además $|u \times v|$ **es el área del paralelogramo determinado por los dos vectores.**

Producto cruz entre vectores

El signo del producto indica la orientación de los vectores.



Además, notemos que aquí $u \times v = 0$ significa que los dos vectores tienen igual dirección (¡no confundir con el producto escalar!).

Contenidos

¿Cómo represento una recta?

Una recta puede ser representada de más de una forma: las dos más usuales son:

- $L(x) = mx + b$, siendo m la pendiente de la recta y b la ordenada al origen.
- $L(t) = \mathbf{p} + t\mathbf{v}$, siendo $\mathbf{v} \in \mathbb{R}^2$ el vector dirección y $\mathbf{p}$ un punto en la recta. $t \in \mathbb{R}$ es un escalar que representa cuánto se dilata o contrae el vector $\mathbf{v}$.

El primer enfoque tiene una posible desventaja, ¿cuál es?

¿Cómo represento una recta?

Una recta puede ser representada de más de una forma: las dos más usuales son:

- $L(x) = mx + b$, siendo m la pendiente de la recta y b la ordenada al origen.
- $L(t) = \mathbf{p} + t\mathbf{v}$, siendo $\mathbf{v} \in \mathbb{R}^2$ el vector dirección y $\mathbf{p}$ un punto en la recta. $t \in \mathbb{R}$ es un escalar que representa cuánto se dilata o contrae el vector $\mathbf{v}$.

El primer enfoque tiene una posible desventaja, ¿cuál es?

La pendiente de una recta es un cociente entre dos números, y por ende debemos asegurarnos que no se indefina dicho cociente. Así, si se quiere representar la recta vertical $x = c$, se la deberá tratar como un caso especial en el código (¡Oh no, un *if*!).

Entonces, utilizaremos la representación $L(t) = p + tv$. ¿Cómo la obtenemos a partir de dos puntos que pasen por la recta L?

Entonces, utilizaremos la representación $L(t) = p + tv$. ¿Cómo la obtenemos a partir de dos puntos que pasen por la recta L?

Si a y b se encuentran sobre L, el vector $v = b - a$ representa la dirección de la recta. Un punto p posible será el $p = a$. Así, la recta se escribirá como:

$$L(t) = a + t(b - a) \text{ con } t \in \mathbb{R}.$$

Una gran ventaja de esta forma es que funciona exactamente igual en dimensiones superiores.
Además, si bien a la hora de computar utilizamos necesariamente algún sistema de coordenadas, una enorme ventaja de los vectores es que son entidades *independientes de las coordenadas* (por eso no hay casos especiales con vectores verticales, etc.).

A partir de esto, podemos representar fácilmente un **segmento de recta** (por ejemplo, la sección de L entre a y b). ¿Cómo?

A partir de esto, podemos representar fácilmente un **segmento de recta** (por ejemplo, la sección de L entre a y b). ¿Cómo?

¡Aprovechándonos de t!

- Si $t = 0$, $L(0) = a$.
- Si $t = 1$, $L(1) = b$.
- Si $0 < t < 1$, $L(t)$ es parte del segmento determinado por a y b.
- Si $t < 0$ o $t > 1$, $L(t)$ no será parte del segmento.

Entonces, $L(t)$ pertenece al segmento ya dicho si y sólo si $t \in [0, 1]$.

Contenidos

Intersección de dos rectas

- ¿Qué pasa si tenemos que calcular el punto de intersección de dos rectas en el plano?
- Dependerá de la representación usada, pero concentrémonos en la representación que recomendamos, $\mathbf{p} + t\mathbf{v}$.

Intersección de dos rectas

- ¿Qué pasa si tenemos que calcular el punto de intersección de dos rectas en el plano?
- Dependerá de la representación usada, pero concentrémonos en la representación que recomendamos, $\mathbf{p} + t\mathbf{v}$.
- Hay más de una manera de llegar a la cuenta correcta: veamos la que personalmente más me gusta.

El α-β

$$p_1 + \alpha v_1 = p_2 + \beta v_2$$

Queremos eliminar una de las dos variables desconocidas(α y β). Hagamos entonces de ambos lados producto cruz con v_2:

$$(p_1 + \alpha v_1) \times v_2 = (p_2 + \beta v_2) \times v_2$$

$$p_1 \times v_2 + \alpha v_1 \times v_2 = p_2 \times v_2$$

$$\alpha v_1 \times v_2 = (p_2 - p_1) \times v_2$$

$$\alpha = \frac{(p_2 - p_1) \times v_2}{v_1 \times v_2}$$

Una vez que tenemos α, el punto buscado es claramente $p_1 + \alpha v_1$

El α-β (observaciones)

- Notar que el denominador es $v_1 \times v_2$, y por lo tanto se anula exactamente cuando las rectas son paralelas (vectores v_1 y v_2 alineados).
- Si todas las coordenadas con las que trabajamos son enteras, la única operación que se sale de los enteros es la división para obtener α. Esto prueba de paso que las intersecciones de rectas en estos casos tendrán coordenadas **racionales**.

Contenidos

¿Qué es sweep line?

- La técnica de *sweep line* (y similares) se caracteriza por tener una línea (vertical u horizontal generalmente, pero no necesariamente) que va barriendo todo el plano y recolectando información sobre puntos de él.
- Si se recuerdan ciertos *eventos* que van sucediendo a lo largo del barrido y se los analiza, se pueden obtener soluciones de una complejidad mejor que con otras técnicas.
- **Los eventos serán instantes del barrido en los que sucede algo relevante para el problema.**

En algún sentido, las técnicas de barrido son como la programación dinámica: uno aprende la idea general, pero luego en cada problema se aplica y se utiliza de manera diferente, adaptando la idea general ya vista.

Es por eso que creemos que la mejor manera de aprender estas técnicas es viendo algunos ejemplos de problemas relevantes en los que se las utiliza.

Par de puntos más cercano

Problema

Dados n puntos en el plano, queremos encontrar dos tales que la distancia entre ellos sea la mínima posible (o sea, el par más cercano)

¿Ideas?

Par de puntos más cercano

Problema

Dados n puntos en el plano, queremos encontrar dos tales que la distancia entre ellos sea la mínima posible (o sea, el par más cercano)

¿Ideas? ¿Cómo sé qué idea puede funcionar y cuál no, si no sé cuánto vale n? Si $n \leq 1000$, el problema a resolver es muy distinto que si $n \leq 1000000$.

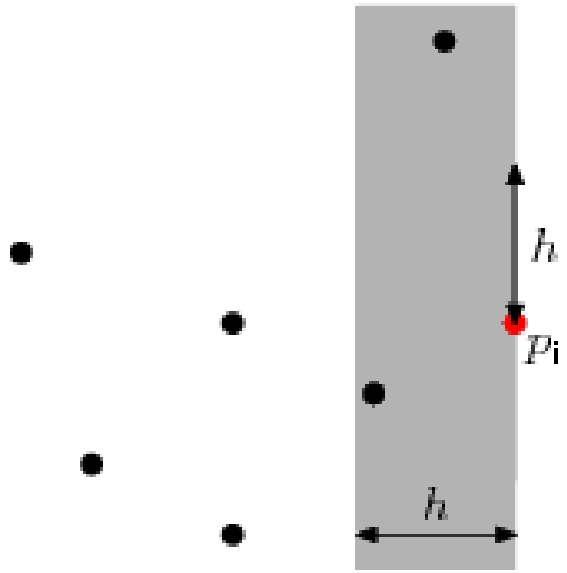
- Existe una solución trivial en $O(n^2)$, que es simplemente mirar cada par y elegir el de menor distancia entre todos los posibles.
- Es una solución muy fácil de implementar, así que si el problema es con $n \leq 1000$ definitivamente deberíamos programar esa.
- ¿Pero y si $n \leq 1000000$? Utilizaremos sweep line para encontrar una solución de mejor complejidad.

- El problema con el enfoque $O(n^2)$ es que a cada punto lo comparamos con **todos** los demás. Tal vez podemos aprovecharnos de la posición en el plano de cada uno para no tener que hacer esto cada vez.
- Barreremos el plano con una recta vertical, de izquierda a derecha.
- Para ser coherentes con este barrido, ordenaremos los puntos crecientemente según su coordenada x. Esto tendrá una complejidad de $O(n \lg n)$.

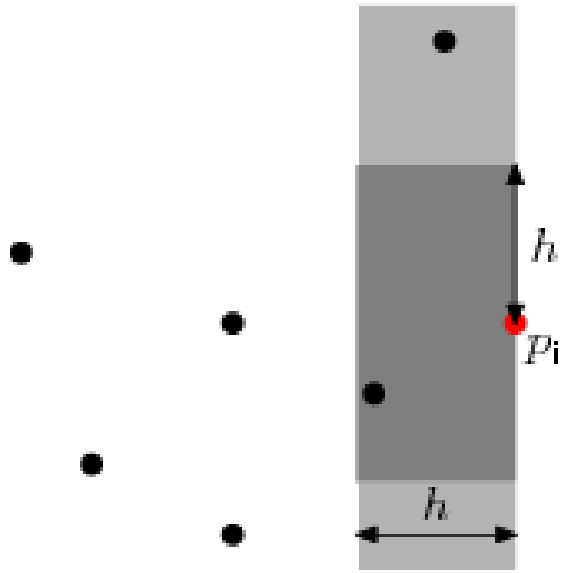
- Llamemos h a la menor distancia entre dos puntos encontrada hasta el momento. La inicializaremos en ∞ (en la práctica, un valor tan grande tal que sea imposible que haya un par de puntos tan lejanos).

- Supongamos que ya pasamos por los primeros $i - 1$ puntos con la línea vertical, y ahora chocamos al i-ésimo punto.

Mantendremos un set con los puntos (ya procesados) cuya coordenada *x* esté a una distancia menor a *h*. Estos puntos son los únicos candidatos a estar a distancia menor que *h* del punto *i*-ésimo, y por ende a ser una mejor solución que la actual.



Observemos también que no cualquier punto cuya coordenada *x* esté a distancia menor a *h* es un candidato real a ser una mejor solución. Si la coordenada *y* de un punto está a una distancia mayor a *h*, tampoco será un candidato.



Así, sólo deberíamos quedarnos con los puntos cuya coordenada y esté en el intervalo $(y_i - h, y_i + h)$.

¿Cómo descartamos los puntos con coordenada x mayor a h, si tenemos a los candidatos en un set?

Así, sólo deberíamos quedarnos con los puntos cuya coordenada y esté en el intervalo $(y_i - h, y_i + h)$.

¿Cómo descartamos los puntos con coordenada x mayor a h, si tenemos a los candidatos en un set?

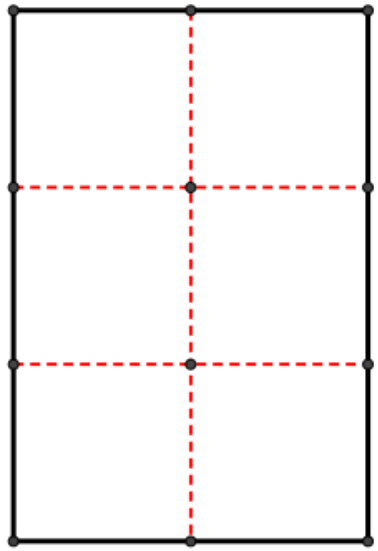
¡Manteniendo el set ordenado por su coordenada y!
Así, *quitar* los puntos cuya coordenada y está demasiado lejos de p_i se reduce simplemente a no mirar los puntos fuera de un intervalo.

Y ahora, para todos los puntos que quedaron en el set calculamos su distancia con p_i y actualizamos h.
¡¿Pero cómo?! Potencialmente, hay muchos puntos en el set.

Y ahora, para todos los puntos que quedaron en el set calculamos su distancia con p_i y actualizamos h.
¡¿Pero cómo?! Potencialmente, hay muchos puntos en el set.

Esto no es tan así. Veamos que quedaron a lo sumo 6 candidatos (6 elementos en el set).

La observación importante es que todos los puntos anteriores están a una distancia mayor o igual a h entre sí, pues si no, la mínima distancia ya habría sido actualizada anteriormente.



No puede haber 2 puntos en el mismo cuadradito, pues luego esos 2 estarían a una distancia menor que h. Así, hay a lo sumo 6 puntos para mirar.

Resumen

El algoritmo consta de 3 pasos importantes:

1. Ordenar los puntos por su coordenada x (para poder hacer el barrido con una línea vertical). Esto toma $O(n \lg n)$.
2. Insertar y remover cada punto una vez del set ordenado por coordenada y, que toma $O(n \lg n)$ (insertar y remover un elemento toma $O(\lg n)$ usando un set, y hay n puntos en total).
3. Comparar cada punto con una cantidad constante de candidatos (≤ 6), lo que toma $O(n)$ en total.

Así, en total el algoritmo es $O(n \lg n)$!

Problemas para pensar

http://www.spoj.com/problems/CLOSEST/
http://goo.gl/UT0O2U (A Safe Bet - WF 2012)

Contenidos

¿Qué es sweep circle?

- La idea de sweep circle es exactamente la misma que la ya mencionada sweep line, pero en lugar de mover una recta imaginaria, movemos una circunferencia.
- La circunferencia puede moverse en linea recta (traslación) o alrededor de un centro fijo (rotación).
- Como un círculo es una figura acotada, el choque de la circunferencia con los puntos interesantes produce eventos de *entrada* y de *salida* en el círculo.

Ejemplo: Ubicación ideal de un círculo en el eje Y

Problema

Dado un radio $R > 0$ entero, se debe indicar cuál es la máxima cantidad de puntos de la grilla de coordenadas enteras que es posible encerrar con un círculo de radio R, cuyo centro se encuentre posicionado sobre la recta $x = 0$ (el eje y).

Ejemplo: Ubicación ideal de un círculo en el eje Y

Problema

Dado un radio $R > 0$ entero, se debe indicar cuál es la máxima cantidad de puntos de la grilla de coordenadas enteras que es posible encerrar con un círculo de radio R, cuyo centro se encuentre posicionado sobre la recta $x = 0$ (el eje y).

Observación: Alcanza con considerar las posiciones $0 \leq y \leq 1$

Planteo con sweep circle

- Comenzamos con el círculo ubicado en $(0, 0)$, y todos los correspondientes puntos de la grilla adentro.
- "Movemos" el círculo en vertical, hasta llegar a $(0, 1)$, procesando los eventos de entrada y salida de puntos.
- La máxima cantidad de puntos que tengamos dentro del círculo en cualquier momento, es el resultado.

Planteo con sweep circle

- Comenzamos con el círculo ubicado en $(0, 0)$, y todos los correspondientes puntos de la grilla adentro.
- "Movemos" el círculo en vertical, hasta llegar a $(0, 1)$, procesando los eventos de entrada y salida de puntos.
- La máxima cantidad de puntos que tengamos dentro del círculo en cualquier momento, es el resultado.

- Notar que hay solamente $O(R)$ eventos de entrada / salida, y además la cantidad de puntos totales dentro del círculo inicial puede computarse en $O(R)$.
- Con todo esto y la técnica de barrido, el problema se resuelve en $O(R \lg R)$

Contenidos

Cápsula convexa (Convex Hull)

Problema

Se tiene un millón vacas en un campo. Se quieren encerrar todas las vacas con una cerca, de manera tal que se minimice la cantidad de alambre utilizado (las vacas deben quedar en el borde o dentro de la cerca).

¿Ideas?

Cápsula convexa (Convex Hull)

Problema

Se tiene un millón vacas en un campo. Se quieren encerrar todas las vacas con una cerca, de manera tal que se minimice la cantidad de alambre utilizado (las vacas deben quedar en el borde o dentro de la cerca).

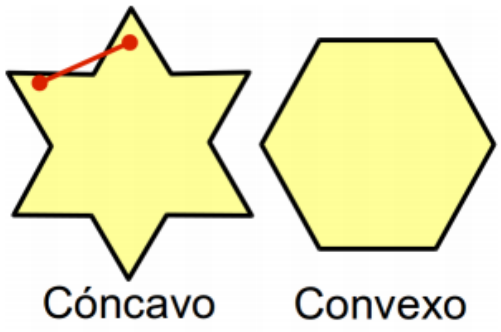
¿Ideas?

Para resolver el problema, tendremos que introducir dos conceptos antes.

Polígonos convexos

Polígono convexo

Un polígono es convexo si cumple que dados dos puntos cualesquiera en su interior, el segmento que los une está completamente contenido en él.

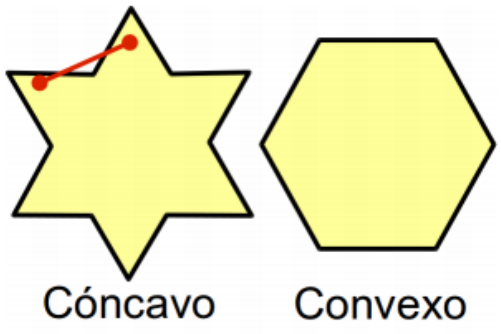


Equivalentemente, un polígono es convexo si los todos sus ángulos interiores son menores que 180°.

Polígonos convexos

Polígono convexo

Un polígono es convexo si cumple que dados dos puntos cualesquiera en su interior, el segmento que los une está completamente contenido en él.

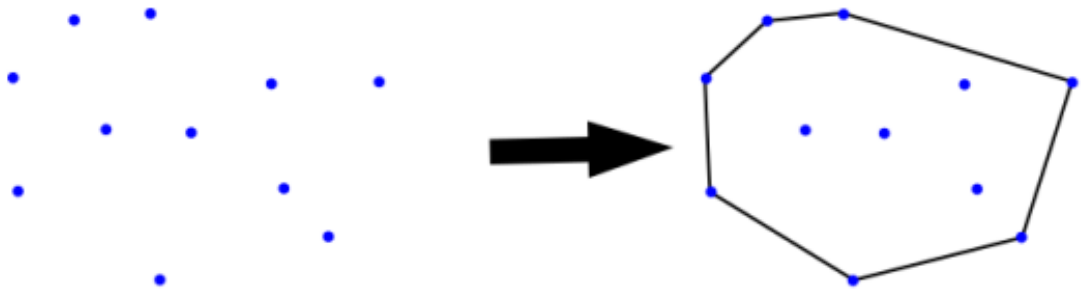


Equivalentemente, un polígono es convexo si los todos sus ángulos interiores son menores que 180°.

Notar que el polígono que será solución al problema debe ser convexo.

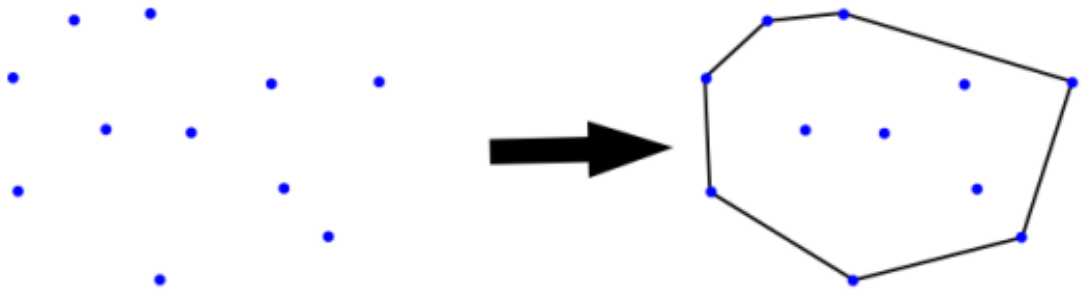
Cápsula convexa

Dados *n* puntos, la *cápsula convexa* es el menor polígono convexo que contiene a todos los puntos en su interior. Se puede probar que es única.



Cápsula convexa

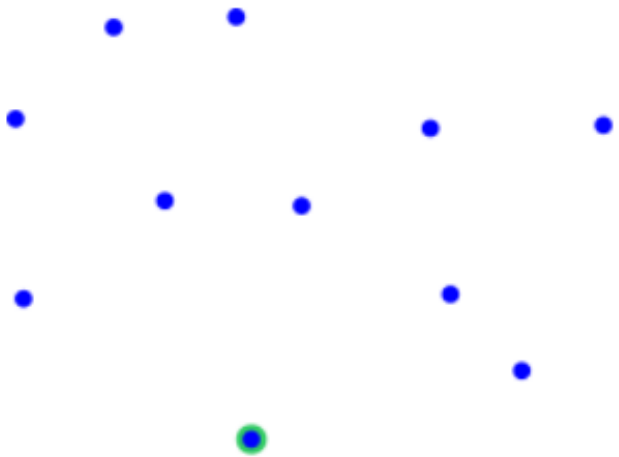
Dados *n* puntos, la *cápsula convexa* es el menor polígono convexo que contiene a todos los puntos en su interior. Se puede probar que es única.



Entonces, podemos ver que el problema se reduce a hallar la cápsula convexa. ¿Pero cómo lo hacemos?

Algoritmo de Graham Scan

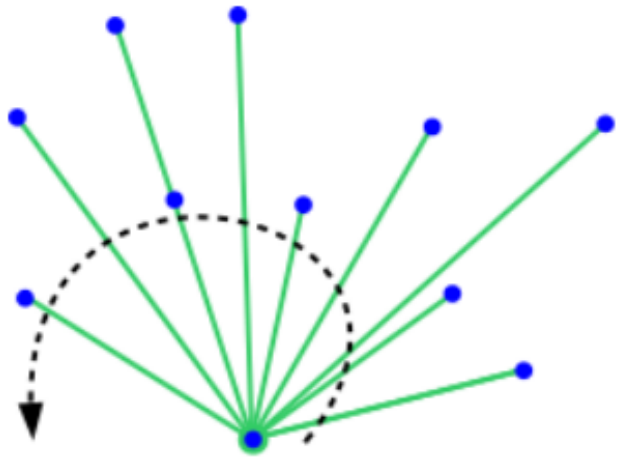
Se elige el punto de más abajo (con menor y), y en caso de haber más de un punto con la misma y se elige el punto de más a la izquierda (o sea, con menor x). A este punto lo llamaremos P. Elegirlo tiene una complejidad $O(n)$.



Iremos barriendo el plano con una semirrecta que parta desde P. Esta semirrecta no será vertical ni horizontal, como veníamos viendo, sino que **girará en sentido antihorario**.

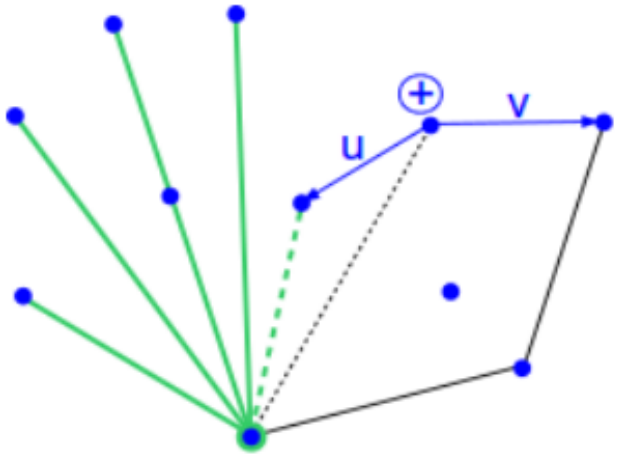
Iremos barriendo el plano con una semirrecta que parta desde P. Esta semirrecta no será vertical ni horizontal, como veníamos viendo, sino que **girará en sentido antihorario**.

Para poder hacerlo, tenemos que ordenar los puntos por su ángulo respecto a P y al eje x. Si dos puntos tienen el mismo ángulo, desempatamos por el más cercano primero.

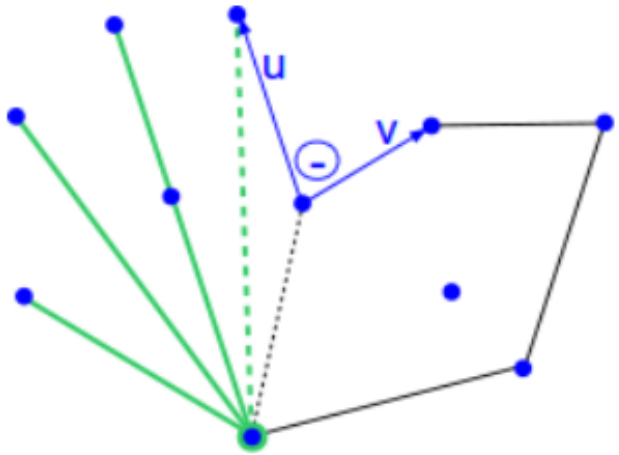


Ahora veamos cómo manejamos los eventos (en este caso, será simplemente la aparición de un punto en el barrido).

El próximo punto siempre está en la Convex Hull. En este caso, $u \times v > 0$.



Mientras que $u \times v < 0$, sacamos el punto anterior (¡forma un ángulo mayor que 180°!).



Pseudocódigo del Graham Scan

```
1  vector<Punto> ConvexHull ( vector<Punto>& lista ) {
2      if( |lista| < 3 ) devolver lista
3      p = el punto de mas abajo y mas a la izquierda
4      Ordenar todos los puntos por angulo respecto a p
5          (desempatando por distancia a p, los mas cercanos primero)
6      pila = stack de puntos vacio
7      pila.push(lista[0]) // lista[0] == p
8      pila.push(lista[1])
9      i = 2
10     while (i < N)
11         sea a el tope de pila, b el anteultimo de la pila (si existen)
12         if (pila.size > 1 && pcruz(lista[i] - a , b - a) <= 0 )
13             pila.pop()
14         else
15             pila.push(lista[i])
16             i = i + 1
17     devolver pila
18 }
```

Problemas para pensar

- goo.gl/rT7Ji
- goo.gl/lIEHC

Referencias

- *Introduction to Algorithms, 2nd Edition*. MIT Press.
  Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein
  Sección 33 (Computational Geometry)
- `https://www.topcoder.com/tc?module=Static&d1=tutorials&d2=geometry1`
- `https://www.topcoder.com/tc?module=Static&d1=tutorials&d2=geometry2`
- `https://www.topcoder.com/tc?module=Static&d1=tutorials&d2=geometry3`
- `https://www.topcoder.com/tc?module=Static&d1=tutorials&d2=lineSweep`

Contenidos

Grafo planar

Definición

Un grafo se dice *planar* si es posible dibujarlo en el plano, haciendo corresponder a cada vértice un punto, y a cada arista una curva simple continua que una los puntos correspondientes a los extremos de la arista, de manera tal que dos curvas correspondientes a aristas distintas no se intersequen más que en sus extremos.

Definición

Dado un grafo planar G, a un dibujo de G en el plano que cumple lo enunciado en la definición anterior se lo denomina un *embedding*, *inmersión* o simplemente *dibujo* de G.

Notar que un mismo grafo planar G puede tener infinitos embeddings distintos.

Región

Definición

Dado un embedding E de un grafo planar G, se denomina una *región* de E a una componente conexa del conjunto de puntos del plano que no forman parte del dibujo de G en E.

Notar que al igual que muchas otras propiedades de un dibujo de un grafo planar, el conjunto de regiones depende del dibujo, y **en principio**, distintos dibujos de un mismo grafo planar podrían tener diferente cantidad de regiones.

Región (cont)

Definición

Dada una región f de un dibujo de un grafo planar G, se denomina el *grado* de f y lo notaremos $d(f)$, a la cantidad de aristas presentes en la *frontera* de f en el dibujo. Además, si la región f toca a la arista de ambos lados, entonces será contada dos veces para el grado.

Notar que de la definición surge que cada arista "aporta grado" a exactamente dos regiones (o bien, a una misma región dos veces), de donde siempre se tiene $\sum_f d(f) = 2m$

Contenidos

Fórmula de Euler

La principal herramienta para trabajar con grafos planares es el siguiente resultado:

Teorema

Si G es un grafo planar conexo de n vértices y m aristas, y R es la cantidad de regiones de **cualquier** dibujo de G, entonces:

$$R + n = m + 2 \text{ (fórmula de Euler)}$$

En general, para un grafo planar cualquiera con $c \geq 1$ componentes conexas vale:

$$R + n = m + c + 1$$

Observar que esto es válido incluso si el grafo contiene multiejes (más de un eje entre un mismo par de nodos) y bucles (ejes de un nodo a sí mismo).

Raleza de los grafos planares

Sea G un grafo simple (sin multiejes ni bucles) planar, y g la longitud mínima de un ciclo simple de g (si G no tiene ciclos tendremos directamente $m \leq n-1$).

Teorema

Si G cumple lo anterior, entonces $m \leq \frac{(n-c-1)g}{g-2}$.

Corolario

Si G es grafo simple planar con $n \geq 3$, entonces $m \leq 3n-6$.

Para demostrar esto, notamos que la frontera de una región debe contener un circuito, así que

$$2m = \sum_f d(f) \geq Rg = (m-n+c+1)g \Rightarrow m \leq \frac{(n-c-1)g}{g-2}$$

Ejemplos mínimos de grafos no planares

Como consecuencia de lo anterior, notamos que:

- K_5 no es planar: tiene $m = 10$ y $n = 5$, y no cumple $m \leq 3n - 6$.
- $K_{3,3}$ no es planar: tiene $m = 9$, $n = 6$, $g = 4$ y $c = 1$, y no cumple $m \leq \frac{(n-c-1)g}{g-2} = \frac{(6-1-1)4}{2} = 8$.

Estos son los ejemplos no planares con menor cantidad de nodos y aristas, respectivamente.