UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Ciencias

Autor: Adrián Aguilera Moreno

Geometría Computacional

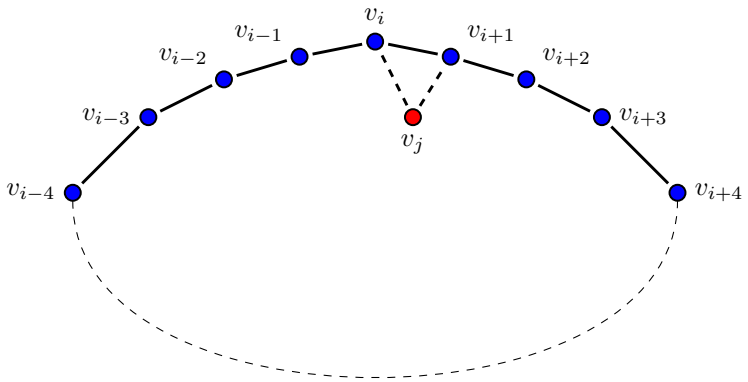
Tarea 01

1. Sea S un conjunto de puntos en el plano en posición general. Demuestra que el cierre convexo de S es el polígono convexo con perímetro y área más pequeños que contienen a S.

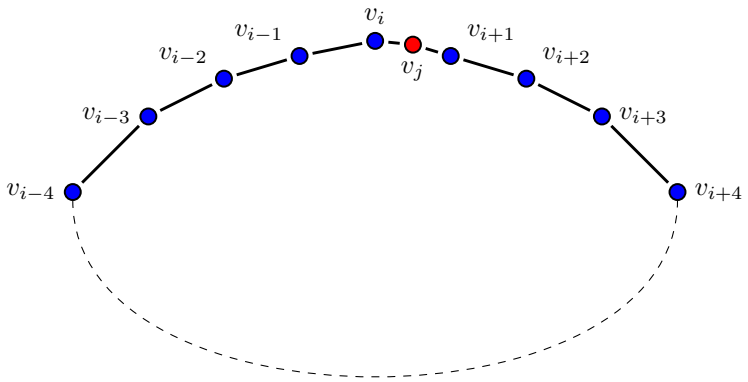
Dem. Para este problema, dividamos la prueba en dos posibles casos:

- Con $C(S)$[1] el cierre convexo. $C(S)$ es el polígono convexo con perímetro más pequeño que contiene a S.
  Prueba por *reducción al absurdo*. Supongamos que $C(S)$ no es el polígono de perímetro mínimo que envuelve todos los puntos en S. La suposición anterior implica que existe $C(S)' \neq C(S)$ tal que $P(C(S)') < P(C(S))$[2] con $C(S)'$ un segundo cierre convexo. Entonces, ¿Cómo modificar $C(S)$ para convertirlo en $C(S)'$? Esto puede ocurrir en 3 casos, estos son:

  Caso 1. Existe un punto interno al polígono que lo convierte en uno de menor perímetro, en este caso perdemos convexidad en la nueva figura y por tanto debemos descartar este caso (pues esta condición es parte del antecedente de nuestra implicación).



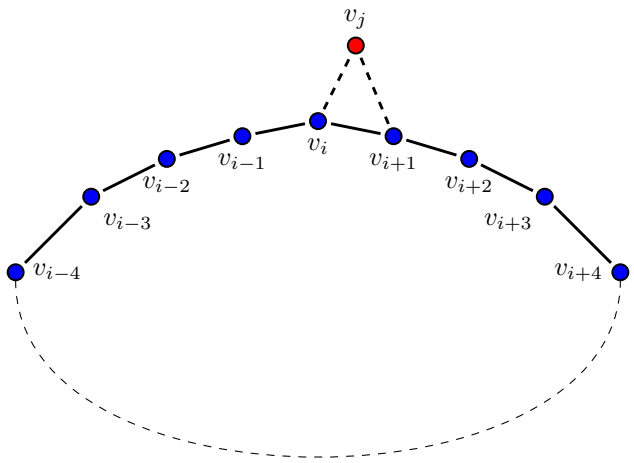
  Caso 2. Existe un punto, extra en $C(S)'$ y que no esta en $C(S)$, sobre un segmento de $C(S)$, esto no reduce el perímetro y por tanto podemos descartar este caso.



[1] Diremos que C es la función que nos devuelve como imagen el cierre convexo del conjunto de puntos que se pase como argumento.

[2] Diremos que P es la función que se mapea al perímitro del polígono que se pasa como argumento.

Caso 3. Este caso lo anexo solo para estar completo, pero no debe suceder por la definición de $C(S)$, pues es el polígono convexo formado por la envolvente convexa del conjunto de puntos en S, así no debe haber un punto que no quede interno de $C(S)$ y en consecuencia de $C(S)'$.



Por lo anterior, hemos llegado a una contradicción en cada caso exhibido. Pues ninguno cumple ser un polígono resultado de la envolvente convexa de menor perímetro que $C(S)$.

$\therefore$ C(S) es de menor perímetro entre los que contienen a S.

- Con $C(S)$ el cierre convexo. $C(S)$ es el polígono convexo con área más pequeña, tal que $C(S)$ contiene a S.
  Prueba por *reducción al absurdo.* Analicemos los casos 1 y 2, anteriores y supongamos que existe $A(C(S)') < A(C(S))$.
  - *Caso 1.* Perdemos convexidad y por tanto llegamos a una contradicción, pues cualquier punto interno nos trae como consecuencia la pérdida de convexidad.
  - Caso 2. El tener un punto en un segmento no disminuye el área del polígono, por tanto contradice que $A(C(S)') < A(C(S))$.

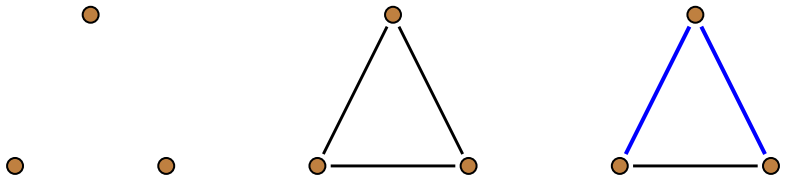
  Por lo anterior, hemos llegado a una contradicción en cada caso exhibido. Pues ninguno cumple ser un polígono resultado de la envolvente convexa de menor área que $C(S)$.

  $\therefore$ C(S) es de menor área entre los que contienen a S.

2. Se define el diámetro de un conjunto de puntos S como la distancia más grande entre cualesquiera dos puntos de S, denotado por d. Demuestra que d está formado por dos vértices del cierre convexo de S.

Dem. Procedamos por inducción en el tamaño de S.

Veamos que pasa cuándo $|S| = 3$. Como podemos notar, el diámetro contiene $3 > 2$ puntos.



Conjunto S. Envolvente convexa. Diámetro de S.

Supongamos que cuando $|S| = k$, el diámetro de S contiene dos puntos del polígono formado por el cierre convexo de S.

¿Qué pasa cuándo nuestro conjunto de puntos S aumenta en exactamente un punto? Existen 2 casos interesantes para dar respuesta a esta pregunta, analicemos estos por separado

- Si el punto esta contenido[3] en $C(S)$, terminamos. Pues, por hipótesis, nuestro diámetro ya contiene dos puntos en $C(S)$ y la anexión del punto distinguido [digamos v] modificaría al diámetro solo para aumentar v (* esto lo podríamos realizar tomando los puntos x y y más cercanos a v que sean parte del diámetro, borar xy de diámetro y añadir xv y vy a este).
- Si el punto esta al exterior de $C(S)$, debemos calcular la envolvente convexa de $C(S')$, donde $S' = S + v$, esto lo podemos realizar econtrando las tangentes a v con respecto de $C(S)$ y unirlas, este sería el nuevo cierre convexo. Lo anterior implica que v será parte del polígono formado por la envolvente convexa (de otro modo, v sería punto interno y se cubre en el caso anterior). Por $(*)$ podemos anexar v en d, y si el punto más cercano a v es un extremo entonces basta con anexar ese segmento a d. Supongamos que cuando realizamos la unión de v con $C(S)$ perdimos los puntos en $C(S)$ que también estaban en d garantizados por la hipótesis, entonces ya tenemos a v y como nadie en la envolvente pertenece a d, podemos anexar al vecino de v en $C(S')$ a d y terminamos.

[3]Casos: el punto esta al interior de $C(S)$, el punto esta en un segmento de $C(S)$.

3. Dado un conjunto de puntos S diseña un algoritmo que encuentre un polígono simple cuyos vértices sean el conjunto S.

Solución. Para el diseño de este algoritmo tomaremos como base el algoritmo `Graham` visto en clase, así, exhibamos el algoritmo

1. Primero debemos encontrar un punto distinguido p_0. Este punto puede ser cualquier punto y aquí nos podemos preguntar ¿Por qué cualquier punto? La respuesta es simple, todos los puntos serán parte de nuestro polígono y por tanto no da exactamente igual con cuál iniciar. Este paso tiene una complejidad contenida en $\mathcal{O}(1)$.

2. Ahora obtengamos un orden angular respecto a p_0, esto nos toma $\mathcal{O}(n \log_2 n)$ por la cota de ordenamiento por comparaciones existente.

3. Ahora que ya tenemos un orden angular y un punto por el cuál iniciar, recorramos nuestros puntos tomando como siguiente, siempre, al próximo en el orden (así creamos una arista en cada iteración en el recorrido y lo guardamos, digamos en una lista) en sentido contrario a las manecillas del reloj[4]. Esto nos simplifica el detalle de conocer nuestras direcciones (validar) para poder regresar en caso de un giro en sentido contrario al requerido en `Graham`. Esto nos toma la cantidad de puntos en S, por tanto tenemos un orden $\mathcal{O}(n)$. Eventualmente llegaremos a p_0 y es en este momento que nuestro algoritmo termina regresando las aristas encontradas durante el recorrido.

¿Por qué será cierto que nuestro algoritmo no encuentra aristas que se intersecten? Por el orden encontrado en (2).

Análisis de complejidad. nuestro algoritmo tiene una complejidad en

$$\therefore \mathcal{O}(1) + \mathcal{O}(n \log_2 n) + \mathcal{O}(n) \in \mathcal{O}(n \log_2 n).$$

[4] ¿Esto necesario? No, es necesario ir en algún orden. No necesariamente este, sin embargo este orden es suficiente.

4. Diseña un algoritmo para encontrar la envolvente convexa de un polígono monótono en tiempo lineal. Un polígono P es monótono respecto a una línea ℓ si cualquier línea perpendicular a ℓ intersecta a P en a lo más dos puntos. Puedes suponer que se conoce ℓ.

Solución. Este problema nos da información extra y valiosa. Esto nos permite modificar el algoritmo de `Graham` para encontrar un orden respecto a los puntos de P en tiempo lineal y reducir nuestra complejidad. A continuación, se exhibe el algoritmo requerido

1. Encontremos un punto distinguido p_0 tal que sea el más "chico" respecto X o Y, por simplicidad este algoritmo asume que p_0 es el más "chico" con respecto a Y. Esto lo encontramos en $\mathcal{O}(n)$.

2. Obtener un orden para poder recorrer. Aprovechando la ℓ-monotonía de nuestro polígono, proyectemos cada punto en ℓ por medio de una transformación $T : (v \in P) \to (v' \in \ell)$. Por tanto, cada proyección nos toma $\mathcal{O}(1)$, como debemos proyectar los n puntos en P entonces gastamos $\mathcal{O}(n)$.

   Ahora, recorramos ℓ (que ya contiene las proyecciones hechas a los puntos en P) por medio de las proyecciones y finalmente obtenemos un orden. Esto nos toma nuevamente $\mathcal{O}(n)$.

   Obs. ¿Qué pasa si existen $u, v \in P$ tal que $T(u) = T(v)$? La respuesta es un poco sencilla, pues basta con encontrar las proyecciones superior e inferior y guardar dos ordenes en cuanto a las proyecciones[5], lo cuál es trivial de obtener al crear una paralela a ℓ, digamos ℓ' tal que P quede entre la manga formada por ℓ y ℓ'. Así, realizamos proyecciones en cada recta (respecto a la cota de inferior y superior respecto de ℓ), recorremos, y obtenemos un orden que podemos fusionar en $\mathcal{O}(1)$. Concluímos que este paso tiene complejidad en $2 \cdot \mathcal{O}(n)$.

3. Por último recorremos el orden en sentido contrario de las manecillas del reloj y recorremos hacia la izquierda, siempre, verificando las direcciones de vuelta. Pues no queremos vueltas a la derecha. Como cada punto conoce a su vecino (por el orden), esto lo podemos hacer en $\mathcal{O}(1)$. Como debemos hacer esto para cada punto, nos toma una complejidad contenida en $\mathcal{O}(n)$. En cada iteración formamos una arista y la guardamos, eventualmente llegaremos a p_0 que es desde dónde inicia nuestro algoritmo.

Análisis de complejidad. Este algoritmo tiene una complejidad en

$$\therefore \mathcal{O}(n) + 2 \cdot \mathcal{O}(n) + \mathcal{O}(n) = 3 \cdot \mathcal{O}(n) \in \mathcal{O}(n).$$

[5]Esto lo decidimos con la coordenada no involcrada en cada de (X, Y, ℓ)-monotonía y para π-monotonía tomamos como parámetro su radio.