Cubiertas convexas

Dr. Eduardo A. RODRÍGUEZ TELLO

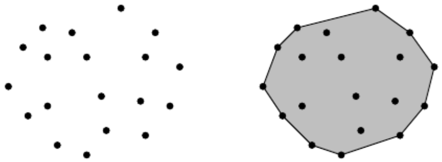
CINVESTAV-Tamaulipas

18 de enero del 2013

1. Cubiertas convexas
   - Introducción
   - El problema de la cubierta convexa
   - Algoritmo simple para cubiertas convexas
   - Algoritmo de Graham para cubiertas convexas
   - Bibliografía
   - Tarea 2

Convexidad

- El día de hoy vamos a estudiar un concepto fundamental en geometría computacional, llamado cubierta convexa
- Intuitivamente, una cubierta convexa puede definirse como una banda elástica que rodea una colección de puntos, la cual se ajusta exactamente al contorno de los puntos (más adelante lo definiremos más formalmente)



Convexidad

- Existen muchas razones por las cuales un cubierta convexa de un conjunto de puntos es una estructura geométrica importante
  - Es una de las aproximaciones de forma de un conjunto de puntos más simples (otras incluyen rectángulos, círculos, etc.)
  - Puede ser usada para aproximar formas más complejas (cubiertas convexas de polígonos o poliedros)
  - Algunos algoritmos calculan la cubierta convexa como una etapa inicial (preprocesamiento) de su ejecución (filtrar puntos irrelevantes)

Convexidad

- Por ejemplo, el diámetro de un conjunto de puntos es la máxima distancia entre cualesquiera dos puntos del conjunto
- Puede demostrarse que el par de puntos que determina el diámetro son ambos vértices de la cubierta convexa
- También se puede observar que las mínimas formas convexas envolventes (rectángulo, círculo, etc.) depende sólo de los puntos de la cubierta convexa

Definiciones formales

- **Convexidad:** un conjunto S es *convexo* si dados cualesquiera dos puntos $p, q \in S$ implican que el segmento de línea $\overline{pq} \subseteq S$
- **Combinación convexa:** dado un conjunto finito de puntos $\{p_1, p_2, \ldots, p_m\}$ una combinación convexa es cualquier punto que puede ser expresado como una suma ponderada,

$$\sum_{i=1}^{m} \alpha_i p_i \quad (1)$$

- donde $0 \leq \alpha_i \leq 1$ y $\sum_{i=1}^{m} \alpha_i = 1$
- Por lo tanto un segmento de línea consiste de todas las combinaciones convexas de sus puntos extremos

Definiciones formales

- **Cubierta convexa:** dado un conjunto de puntos S su cubierta convexa de es el conjunto de puntos que pueden ser expresados como combinaciones convexas de los puntos en S
- Una cubierta convexa de un conjunto de puntos S también puede ser definida como la intersección de todos los conjuntos convexos que contiene S
- Intuitivamente, una cubierta convexa es el conjunto convexo más pequeño que contiene S y se denota como $conv(S)$

Algunos términos de topología

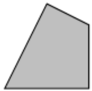
- Para propósitos de este curso requerimos ciertos conceptos de topología
- Estos términos tienen definiciones formales que dejaremos a un lado, por ahora, para dar definiciones más intuitivas
- El *r*-**vecindario** de un punto *p* es el conjunto de puntos que se encuentran a una distancia estrictamente menor a *r* (esfera abierta con centro en *p* y radio *r*)

Algunos términos de topología

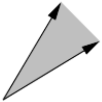
- Dado un conjunto S, un punto p es un **punto interior** de S si la vecindad de p para un radio r está contenida en S
- Un punto es **exterior** si cae en el interior del complemento de S
- Un punto que no es ni interior ni exterior es un punto frontera
- Un conjunto es **abierto** si éste no contiene ninguno de sus puntos frontera

Algunos términos de topología

- Un conjunto es **cerrado** si su complemento es cerrado



- Decimos que un conjunto geométrico está **acotado** si puede ser encerrado en una esfera de radio finito



- Un conjunto es **compacto** si es tanto cerrado como acotado

Algunos términos de topología

- En general, los conjuntos convexos pueden tener fronteras rectas o curvas y pueden ser acotados o no acotados
- Los conjuntos convexos pueden ser topológicamente abiertos o cerrados

- La cubierta convexa de un conjunto finito de puntos en el plano es un polígono acotado, cerrado, convexo

El problema de la cubierta convexa

- El problema (planar) de la cubierta convexa se define de la siguiente forma
- Dado un conjunto P de n puntos en el plano, calcular la representación del polígono convexo cerrado que representa la cubierta convexa de P
- La representación más simple de una cubierta convexa es la enumeración en el sentido inverso a las manecillas del reloj ($\circlearrowleft$) de sus vértices
- Idealmente la cubierta convexa debe consistir sólo de los puntos extremos, en el sentido que si tres puntos caen en un vértice de la frontera de la cubierta convexa, entonces el punto medio no debe ser tomado en cuenta como parte de la cubierta

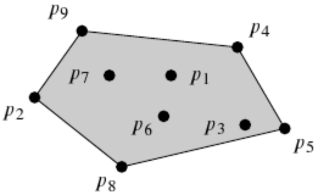
El problema de la cubierta convexa

input = set of points:

$p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9$

output = representation of the convex hull:

p_4, p_5, p_8, p_2, p_9



El problema de la cubierta convexa

- Esto no es un problema si asumimos que los puntos están en la posición general, y en particular no hay tres colineales
- Existe un algoritmo simple para calcular la cubierta convexa en $O(n^3)$, que funciona considerando cada par ordenado de puntos (p, q), y determinado si todos los puntos restantes del conjunto caen dentro del semi-plano que cae a la derecha de línea dirigida de p a q.
- Observe que esto puede ser verificado usando la prueba de la orientación

Algoritmo simple para cubiertas convexas

```
Algorithm SlowConvexHull(P)
Input. A set P of points in the plane.
Output. A list 𝓛 containing the vertices of 𝓒𝓗(P) in clockwise order.
E ← ∅.
for all ordered pairs (p, q) ∈ P × P with p not equal to q
    do valid ← true
       for all points r ∈ P not equal to p or q
           do if r lies to the left of the directed line from p to q
                 then valid ← false.
       if valid then Add the directed edge pq→ to E.
From the set E of edges construct a list 𝓛 of vertices of 𝓒𝓗(P), sorted in clockwise order.
```

- Se verifican $n^2 - n$ pares de puntos. Cada par se compara con otros $n - 2$ puntos, lo que toma $O(n^3)$. El paso final toma $O(n^2)$
- El tiempo total de ejecución es $O(n^3)$, ¿Podrá hacerse más eficientemente?

Algoritmo de Graham para cubiertas convexas

- Ahora presentaremos un algoritmo $O(n \log n)$ para cubiertas convexas llamado algoritmo de Graham o exploración de Graham
- Este algoritmo data de comienzos de los años 70
- El algoritmo está basado en un enfoque de solución común para construir estructuras geométricas llamado *construcción incremental*
- En la construcción incremental los objetos (puntos) se agregan uno a la vez, y la estructura (cubierta convexa) se actualiza con cada nueva inserción

Algoritmo de Graham para cubiertas convexas

- Un punto importante a considerar en los algoritmos incrementales es el orden de la inserción
- Si agregáramos puntos en algún orden arbitrario, necesitaríamos un método para probar si el nuevo punto insertado está dentro de la cubierta convexa
- Simplificaría mucho las cosas el agregar los puntos en algún orden apropiado, en nuestro caso, en orden incremental de la coordenada x
- Esto garantiza que cada nuevo punto agregado está fuera de la cubierta convexa

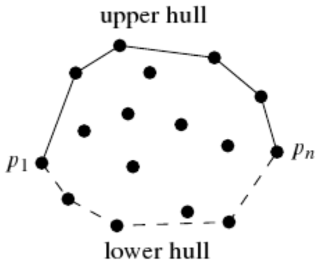
Algoritmo de Graham para cubiertas convexas

- En realidad el algoritmo de Graham original ordenaba los puntos en una manera diferente
- Buscaba el punto más bajo del conjunto de datos y después ordenaba los puntos cíclicamente alrededor de este punto
- Sin embargo, ordenar de acuerdo a la coordenada x es más fácil de implementar

Algoritmo de Graham para cubiertas convexas

- Dado que estamos trabajando de izquierda a derecha (↻), sería conveniente si los vértices de la cubierta convexa también estuvieran ordenados de izquierda a derecha
- Las cubiertas convexas son conjuntos ordenados cíclicos
- Este tipo de conjuntos son algo más complicados de trabajar que los conjuntos ordenados linealmente, por esta razón dividiremos la cubierta convexa en dos, una **superior** y una **inferior**

Algoritmo de Graham para cubiertas convexas



- Los puntos comunes p_1 y p_n a ambas cubiertas serán los vértices extremos izquierdo y de derecho de la cubierta convexa

Algoritmo de Graham para cubiertas convexas

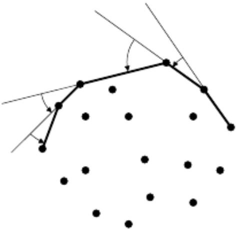
- Después de construir ambas cubiertas, éstas se pueden concatenar en una lista cíclica de izquierda a derecha (↺)
- Como será común durante el cuatrimestre, haremos la suposición de que los puntos están en la posición general
- En este caso significa que ningún par de puntos tienen la misma coordenada x y además que no hay tres puntos colineales

Algoritmo de Graham para cubiertas convexas

- El algoritmo de Graham utiliza una pila (*stack*) H durante su ejecución
- En esta pila el tope o cima (top) corresponde al punto agregado más recientemente
- Vamos a denotar $H.first$ y $H.second$ el tope y el segundo elemento de la pila H respectivamente

Algoritmo de Graham para cubiertas convexas

- Observemos que mientras se leen los elementos de la pila del tope hacia abajo (i.e. de derecha a izquierda) las tripletas consecutivas de puntos de la cubierta superior harán un giro (estricto) hacia la izquierda
- Es decir, tendrán una orientación positiva (↺)

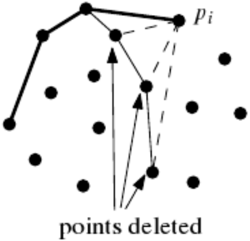


Algoritmo de Graham para cubiertas convexas

- El algoritmo de Graham trabaja de la siguiente forma
- Sea p_i el próximo punto que se agregará al ordenamiento de izquierda a derecha de los puntos
- Si la tripleta $\langle p_i, H.first, H.second \rangle$ tiene orientación positiva, entonces podemos simplemente agregar p_i a la pila

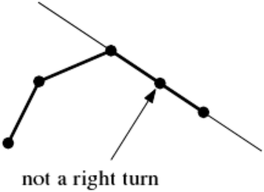
Algoritmo de Graham para cubiertas convexas

- Sino, se pude inferir que el punto medio de la tripleta *H.first* no puede estar en la cubierta convexa
- Por lo tanto lo borramos de la pila



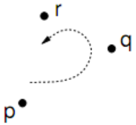
Algoritmo de Graham para cubiertas convexas

- Ésto es repetido hasta alcanzar una tripleta con orientación positiva, o haya menos de dos elementos en la pila

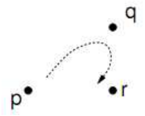


Algoritmo de Graham para cubiertas convexas

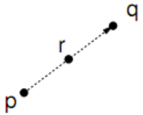
$Orient(p, q, r) > 0$



$Orient(p, q, r) < 0$



$Orient(p, q, r) = 0$



Algoritmo de Graham para cubiertas convexas

Algorithm CONVEXHULL(P)
Input. A set P of points in the plane.
Output. A list containing the vertices of $\mathcal{CH}(P)$ in clockwise order.

1. Sort the points by x-coordinate, resulting in a sequence $p_1, \ldots, p_n$.
2. Put the points p_1 and p_2 in a list $\mathcal{L}_{\text{upper}}$, with p_1 as the first point.
3. **for** $i \leftarrow 3$ **to** n
4.   **do** Append p_i to $\mathcal{L}_{\text{upper}}$.
5.   **while** $\mathcal{L}_{\text{upper}}$ contains more than two points **and** the last three points in $\mathcal{L}_{\text{upper}}$ do not make a right turn
6.   **do** Delete the middle of the last three points from $\mathcal{L}_{\text{upper}}$.
7. Put the points p_n and p_{n-1} in a list $\mathcal{L}_{\text{lower}}$, with p_n as the first point.
8. **for** $i \leftarrow n-2$ **downto** 1
9.   **do** Append p_i to $\mathcal{L}_{\text{lower}}$.
10.   **while** $\mathcal{L}_{\text{lower}}$ contains more than 2 points **and** the last three points in $\mathcal{L}_{\text{lower}}$ do not make a right turn
11.   **do** Delete the middle of the last three points from $\mathcal{L}_{\text{lower}}$.
12. Remove the first and the last point from $\mathcal{L}_{\text{lower}}$ to avoid duplication of the points where the upper and lower hull meet.
13. Append $\mathcal{L}_{\text{lower}}$ to $\mathcal{L}_{\text{upper}}$, and call the resulting list $\mathcal{L}$.
14. **return** $\mathcal{L}$

Bibliografía

Libros de texto

- Mark de Berg, Otfried Cheong, Marc van Kreveld and Mark Overmars. *Computational Geometry: Algorithms and Applications*. Springer, 3rd edition (April 16, 2008), ISBN-10: 3540779736.
- Joseph O'Rourke. *Computational Geometry in C*. Cambridge University Press; 2000 edition (February 15, 2001), ISBN-10: 0521649765.

- Ambos libros los pueden encontrar en la siguiente URL:
- `http://www.tamps.cinvestav.mx/~ertello/gc/ComputationalGeometry.tgz`

Tarea 2

- Implementar en Java los dos algoritmos para cubiertas convexas vistos en clase, con una interfaz gráfica que permita visualizar los puntos y la cubierta resultante
- Preparar un reporte donde se efectúe un análisis de ambos algoritmos en cuanto a su desempeño con respecto al escalamiento del tamaño de las instancias de prueba
- Efectuar el análisis formal de la complejidad computacional del algoritmo de Graham