Localización de puntos en el plano

Dr. Eduardo A. RODRÍGUEZ TELLO

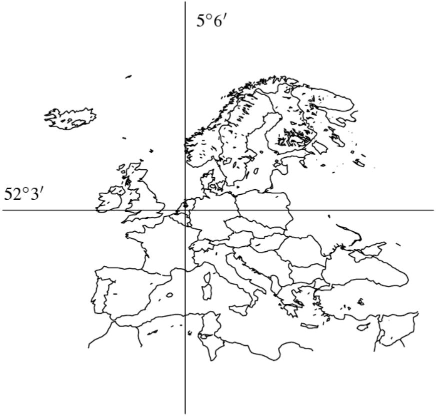
CINVESTAV-Tamaulipas

12 de marzo del 2013

- El material de esta clase está basado en el capítulo 6 del libro: 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.

1. Localización de puntos en el plano
   - Introducción
   - Mapas trapezoidales
   - Construcción de mapas trapezoidales
   - Análisis de complejidad
   - Estructura de datos para localización de puntos
   - Construcción incremental
   - Intersección de segmentos de línea (algoritmo aleatorizado)

Introducción



- Dado un mapa (una subdivisión planar) y la consulta de un punto q encontrar la región del mapa que contiene a q

Introducción

- De manera más formal, el problema de localización de puntos en el plano es el siguiente
- Dada una subdivisión poligonal del plano con n vértices, preprocesar esta subdivisión a fin de que, dada la consulta de un punto q, podamos determinar de manera eficiente que cara de la subdivisión contiene a q
- Podemos asumir que cada cara tiene un identificador, que será devuelto como respuesta

Cinvestav

Introducción

- En general q podría coincidir con una arista o con un vértice
- Pero para simplificar, vamos a suponer que esto no ocurre, aunque estos casos especiales no son tan difíciles de manejar (ver libro de Berg et al.)
- Nuestro objetivo es desarrollar una estructura de datos $O(n)$ en espacio que pueda responder a las preguntas en tiempo $O(\log n)$
- Durante muchos años los mejores métodos conocidos tenían un factor adicional log, ya sea en espacio o en el tiempo de consulta

Introducción

- Kirkpatrick logró un gran avance cuando presentó un algoritmo óptimo en tiempo y espacio
- El algoritmo de Kirkpatrick tenía la desventaja de utilizar factores constantes bastante elevados
- Desde entonces algoritmos óptimos más simples y prácticos se han descubierto

Introducción

- Vamos a presentar tal vez el más sencillo y práctico de los algoritmos óptimos conocidos
- El método se basa en una construcción incremental aleatorizada, la misma técnica utilizada en nuestro algoritmo de programación lineal de la clase pasada

Cinvestav

1. Localización de puntos en el plano
   - Introducción
   - Mapas trapezoidales
   - Construcción de mapas trapezoidales
   - Análisis de complejidad
   - Estructura de datos para localización de puntos
   - Construcción incremental
   - Intersección de segmentos de línea (algoritmo aleatorizado)

Cinvestav

Mapas trapezoidales

- El algoritmo que vamos a presentar se basa en una estructura llamada **mapa trapezoidal**, que recibe también algunos otros nombres en la literatura de geometría computacional
- Como se mencionó anteriormente, normalmente la entrada para un algoritmo de localización de puntos es una subdivisión poligonal planar
- Pero para simplificar este problema utilizaremos la suposición de que la entrada es simplemente un conjunto de segmentos de línea no verticales, denotado $S = (s_1, s_2, \ldots, s_n)$

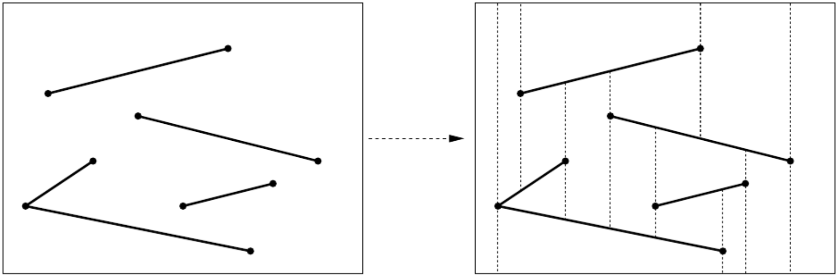
Mapas trapezoidales

- Asumimos también que estos segmentos de línea no se intersectan, salvo posiblemente en sus extremos
- En este contexto, la respuesta a una consulta de la ubicación de un punto es el índice del segmento de línea de S situada inmediatamente debajo del punto de consulta (o algún valor especial si ninguna línea está por debajo del punto de consulta)
- Evidentemente, la formulación original del problema de localización de un punto puede ser fácilmente redefinido de esta manera

Mapas trapezoidales

- Para definir el *mapa trapezoidal*, se dibujan líneas verticales hacia arriba y abajo de cada uno de los vértices de la subdivisión planar (*extensión vertical superior e inferior*)
- Estas líneas se prolongan hasta que hagan contacto con algún otro segmento de S como se muestra en la siguiente figura
- Las líneas trazadas junto con los segmentos originales definen el mapa trapezoidal
- Para evitar que las líneas dibujadas se prolonguen infinitamente asumiremos que la subdivisión original está contenida en un rectángulo (*bounding box*)

Mapas trapezoidales



Mapas trapezoidales

- Además de asumir que no hay segmentos verticales, también se asumirá que no hay dos puntos extremos diferentes que tengan la misma coordenada x
- Esto es para simplificar la descripción del algoritmo
- Aunque las caras de esta subdivisión son de forma trapezoidal, es posible que estén arbitrariamente delimitadas por un gran número de vértices y aristas resultantes de los vértices de otros trapezoides al rededor

Mapas trapezoidales

- Observe que todas las caras de la subdivisión resultante son trapezoides con lados verticales
- El lado izquierdo o derecho puede degenerar en un segmento de línea de longitud cero, lo que implica que el trapezoide resultante degenera en un triángulo
- El proceso de convertir una subdivisión poligonal arbitraria en una descomposición trapezoidales aumenta su tamaño en a lo más un factor constante

Mapas trapezoidales

- Esto se desprende del hecho de que sólo aumentamos el número de vértices por un factor constante y de que el grafo es planar
- Pero debido a que un factor constante en espacio es importante, es una buena idea trabajar esto con cuidado

Mapas trapezoidales

Lema

Dada una subdivisión poligonal con n segmentos, el mapa trapezoidal resultante tiene a lo más $6n + 4$ vértices y $3n + 1$ trapezoides

Demostración

- Para probar la cota sobre el número de vértices, observemos que de cada vértice surgen dos segmentos, cada uno de los cuales dará lugar a la creación de un nuevo vértice
- Así que cada vértice inicial da lugar a tres vértices en el mapa final
- Dado que cada segmento tiene dos vértices, esto implica como máximo $6n$ vértices

Mapas trapezoidales

Demostración...

- Para la cota del número de trapezoides, observe que por cada trapezoide en el mapa final, su lado izquierdo (y el derecho también) está limitado por un vértice de la subdivisión poligonal original
- El extremo izquierdo de cada segmento de línea puede servir como el vértice izquierdo en la frontera de dos trapezoides (uno por arriba del segmento de línea y otro por debajo)
- El extremo derecho de un segmento de línea puede servir por su parte como el vértice izquierdo en la frontera de un trapezoide

Mapas trapezoidales

Demostración...

- Así, cada segmento de la subdivisión original da lugar a como máximo tres trapezoides, para un total de $3n$ trapezoides
- El último trapezoide es aquel limitado por el lado izquierdo del rectángulo (*bounding box*).

Mapas trapezoidales

- Un hecho importante a observar sobre cada trapezoide es que éste se define (i.e., su existencia está determinada) por exactamente 4 elementos de la subdivisión:
  1. un segmento en la parte superior
  2. un segmento en la parte baja
  3. un vértice izquierdo en la frontera
  4. un vértice derecho en la frontera
- Esta simple observación desempeñará un papel importante más adelante en el análisis

Mapas trapezoidales

- Las descomposiciones trapezoidales, como las triangulaciones, son estructuras de datos interesantes en sí mismas
- Es otro ejemplo de la idea de convertir una forma compleja en una colección disjunta de objetos más simples
- El hecho de que los lados sean verticales hacen que los trapezoides sean más sencillos que cuadriláteros arbitrarios
- Por último observe que la descomposición trapezoidal es un refinamiento de la subdivisión poligonal original,
- Por lo tanto una vez que sabemos en qué cara del mapa trapezoidal cae el punto de consulta q, conoceremos en qué cara de la subdivisión original se encuentra éste

1. Localización de puntos en el plano
   - Introducción
   - Mapas trapezoidales
   - Construcción de mapas trapezoidales
   - Análisis de complejidad
   - Estructura de datos para localización de puntos
   - Construcción incremental
   - Intersección de segmentos de línea (algoritmo aleatorizado)

Construcción de mapas trapezoidales

- Vamos a construir el mapa trapezoidal mediante un algoritmo incremental aleatorizado, debido principalmente a que el algoritmo de localización de un punto se basa en esta construcción
- El algoritmo incremental aleatorizado comienza con el rectángulo (i.e., un trapezoide) y luego se añaden los segmentos de la subdivisión poligonal uno por uno en orden aleatorio
- A medida que cada segmento se añade, se actualizará el mapa trapezoidal
- Sea S_i el subconjunto que consistente de los primeros i segmentos (seleccionados aleatoriamente), y sea $\mathcal{T}_i$ el mapa trapezoidal resultante

Construcción de mapas trapezoidales

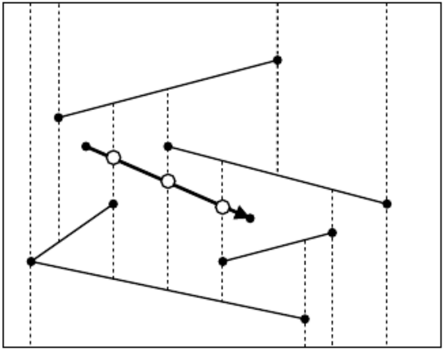
- Para realizar la actualización tenemos que saber en qué trapezoide cae el punto extremo izquierdo del segmento
- Vamos a abordar este punto más tarde, ya que será resuelto por el propio algoritmo de localización de un punto
- Luego se traza el segmento de línea de izquierda a derecha, determinando qué trapezoides intersecta
- Por último, volvemos a estos trapezoides y los reparamos

Construcción de mapas trapezoidales

- Hay dos cosas que están involucradas en la reparación:
  - En primer lugar, los extremos izquierdo y derecho del nuevo segmento requieren que se dibujen extensiones verticales a partir de ellos
  - En segundo lugar, algunas de las primeras extensiones verticales trazadas a partir de los vértices podrían hacer contacto con este segmento de línea. Cuando eso sucede, esas extensiones verticales deben ser recortadas (sabemos cuáles son los vértices de la subdivisión original, por lo que podemos conocer qué lado de la extensión vertical recortar)
- El proceso se ilustra en la siguiente figura

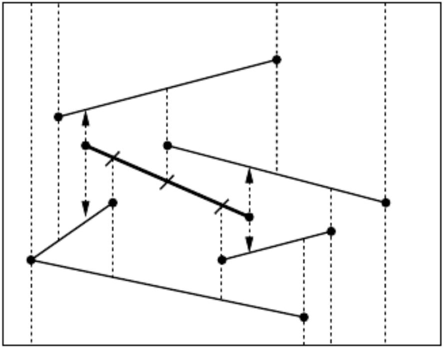
Construcción de mapas trapezoidales

- Localizar los puntos extremos y determinar las intersecciones



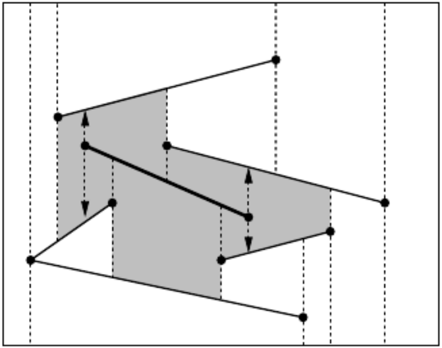
Construcción de mapas trapezoidales

- Trazar las nuevas extensiones verticales y recortar aquellas que intersectan



Construcción de mapas trapezoidales

- Nuevos trapezoides creados



Construcción de mapas trapezoidales

- Observe que la estructura de la descomposición trapezoidal no depende del orden en el que se añaden los segmentos
- Esta observación será importante para el análisis probabilístico
- El siguiente lema es también importante para dicho análisis:

Lema

- Ignorando el tiempo consumido para localizar el punto extremo izquierdo de un segmento, el tiempo que toma insertar el i-ésimo segmento y actualizar el mapa trapezoidal es $O(k_i)$, donde k_i es el número de nuevos trapezoides creados

Cinvestav

Construcción de mapas trapezoidales

Demostración

- Considere la inserción del i-ésimo segmento, donde K denota el número de extensiones verticales que intersecta este segmento
- Tenemos que trazar cuatro extensiones verticales nuevas (dos para cada punto extremo) y luego recortar cada una de las K extensiones verticales existentes, lo que hace un total de $K + 4$ operaciones que deben realizarse
- Si el nuevo segmento no cruza ninguna extensión vertical, entonces obtendríamos exactamente 4 nuevos trapezoides

Construcción de mapas trapezoidales

Demostración...

- Para cada una de las K extensiones verticales que se intersectan, se suma uno más al número de trapezoides nuevos, para un total de $K+4$
- Por lo tanto, sea $k_i = K + 4$ el número de trapezoides creado, el número de operaciones de actualización es exactamente k_i
- Cada una de estas operaciones se puede realizar en tiempo $O(1)$, dada una representación apropiada del mapa trapezoidal como una subdivisión planar
- Por ejemplo, una lista de aristas doblemente conectada

1. Localización de puntos en el plano
   - Introducción
   - Mapas trapezoidales
   - Construcción de mapas trapezoidales
   - **Análisis de complejidad**
   - Estructura de datos para localización de puntos
   - Construcción incremental
   - Intersección de segmentos de línea (algoritmo aleatorizado)

Análisis de complejidad

- Consideremos el tiempo esperado de ejecución que toma construir el mapa trapezoidal
- Más tarde veremos que el tiempo necesario para determinar el trapezoide que contiene el punto extremo izquierdo de cada nuevo segmento es en promedio $O(\log n)$
- Demostraremos que el tiempo esperado promedio para agregar cada nuevo segmento es $O(1)$
- Puesto que hay n inserciones, esto conducirá a una complejidad temporal esperada de $O(n(1 + \log n)) = O(n \log n)$

Análisis de complejidad

- Como de costumbre, esta expectativa es sobre la secuencia de números aleatorios, y no depende de la distribución de los segmentos de línea
- Sabemos que el tamaño del mapa trapezoidal final es $O(n)$
- Además, el tamaño total de la estructura de datos para localización de puntos deberá ser proporcional al número de nuevos trapezoides que se crean con cada inserción
- En el peor caso, cuando agreguemos el i-ésimo segmento, éste podría intersectar una gran parte de los $O(i)$ trapezoides existentes, lo que resultaría en un tamaño total proporcional a $\sum_{i=1}^{n} i = \Theta(n^2)$

Análisis de complejidad

- Sin embargo, la ventaja de la construcción incremental es que esto, en promedio, no sucede muy a menudo
- Mostraremos que en promedio, cada inserción resulta en sólo un número constante de trapezoides creados
- Esto es bastante sorprendente en un principio

Análisis de complejidad

- Evidentemente si los segmentos son cortos, entonces cada uno de ellos muy posiblemente no se intersectará con muchos trapezoides
- Pero si la mayoría de los segmentos son largos, podría parecer que casi cada inserción intersectaría $O(n)$ trapezoides
- Lo que nos salva es que, a pesar de que un segmento largo podría intersectar con muchos trapezoides, éste protege a los sementos posteriores de intersectar con muchos trapezoides

Análisis de complejidad

Lema

- Considere la construcción incremental aleatorizada de un mapa trapezoidal, y sea k_i el número de nuevos trapezoides creados cuando el i-ésimo segmento es agregado. Entonces $E(k_i) = O(1)$, donde la expectativa es tomada sobre todas las permutaciones de los segmentos

Demostración

- El análisis se basará en la técnica de análisis inverso

- Recordemos que este análisis se basa en el análisis del valor esperado suponiendo que la última inserción fue aleatoria

- Sea $\mathcal{T}_i$ el mapa trapezoidal después de la inserción del i-ésimo segmento

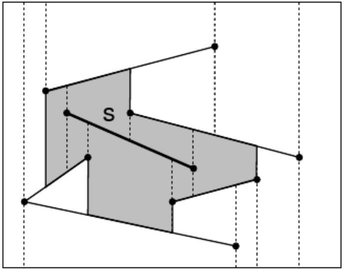
Análisis de complejidad

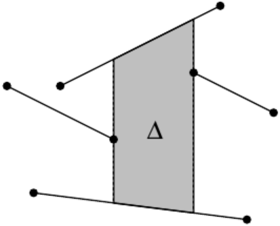
Demostración...

- Debido a que estamos promediando sobre todas las permutaciones entre los i segmentos que están presentes en $\mathcal{T}_i$, cada uno tiene la misma probabilidad $1/i$ de ser el último en ser agregado
- Para cada uno de los segmentos s queremos contar el número de trapezoides que se han creado, siendo s el último segmento en ser agregado
- Supongamos que un trapezoide Δ *depende* de un segmento s, si s ha provocado que Δ se cree, cuando s fue el último segmento agregado

Análisis de complejidad

- a) Trapezoides que dependen de s; b) Segmentos de los que depende el trapezoide Δ





Análisis de complejidad

Demostración...

- Queremos contar el número de trapezoides que dependen de cada segmento, y a continuación calcular la media de todos los segmentos
- Asignemos $\delta(\Delta, s) = 1$ si Δ depende del segmento s, y 0 en caso contrario, entonces el valor esperado es:

$$\begin{aligned} E(k_i) &= \frac{1}{i} \sum_{s \in S_i} (\text{núm. trapezoides que dependen de } s) \\ &= \frac{1}{i} \sum_{s \in S_i} \sum_{\Delta \in \mathcal{T}_i} \delta(\Delta, s) \end{aligned} \quad (1)$$

Cinvestav

Análisis de complejidad

- Algunos segmentos podría haber dado lugar a la creación de numerosos trapezoides y otros a muy pocos
- ¿Cómo podemos llegar a aproximar esta cantidad?
- El truco es, en lugar de contar el número de trapezoides que dependen de cada segmento, contamos el número de segmentos de los cuales cada trapezoide depende
- Esto equivale a la manipulación de invertir el orden de las sumatorias, que es comúnmente usado en combinatoria

Análisis de complejidad

- En otras palabras, podemos expresar la Ecuación 1 como:

$$E(k_i) = \frac{1}{i} \sum_{\Delta \in \mathcal{T}_i} \sum_{s \in S_i} \delta(\Delta, s) \tag{2}$$

- Esta cantidad es mucho más fácil de analizar
- En particular, cada trapezoide está delimitado por un *máximo* de cuatro lados (posibles triángulos)
- Las partes superior e inferior están determinadas cada una por un segmento de S_i, y evidentemente si alguno de estos fue el último en agregarse, entonces este trapezoide se originó como consecuencia de este hecho

Análisis de complejidad

- Los lados izquierdo y derecho están determinados cada uno por un punto extremo de un segmento en S_i, y si alguno de estos fue el último en agregarse, entonces este trapezoide se generó
- Hay un detalle sutil aquí. ¿Qué pasa si múltiples segmentos comparten el mismo punto extremo?
- Tenga en cuenta que el trapezoide es sólo dependiente del primero de esos segmentos que se agregue, ya que este es el segmento que causó que el vértice existiera

Análisis de complejidad

- Notemos igualmente que el mismo segmento que forma el lado superior o inferior también podría proporcionar el punto extremo izquierdo o derecho
- Estas consideraciones sólo disminuyen el número de segmentos de los cuales depende un trapezoide

Análisis de complejidad

- En resumen, cada trapezoide es dependiente de como máximo 4 segmentos, implicando que $\sum_{s \in S_i} \delta(\Delta, s) \leq 4$
- Dado que $\mathcal{T}_i$ consiste de $O(i)$ trapezoides tenemos que:

$$E(k_i) \leq \frac{1}{i} \sum_{\Delta \in \mathcal{T}_i} 4 = \frac{1}{i} 4 |\mathcal{T}_i| = \frac{1}{i} 4 O(i) = O(1) \quad (3)$$

- Dado que el número esperado de nuevos trapezoides creado con cada inserción es $O(1)$, entonces el número total de trapezoides que se crean en todo el proceso es $O(n)$.
- Este hecho es importante para acotar el tiempo total requerido por el algoritmo incremental aleatorizado

Cinvestav

Análisis de complejidad

- La única cuestión que no hemos considerado en la construcción es cómo localizar el trapezoide que contiene el punto extremo izquierdo de cada nuevo segmento agregado
- Vamos a examinar esta cuestión, y de manera más general cómo hacer la localización de un punto a continuación

Estructura de datos para localización de puntos

- La estructura de datos para localización de puntos se basa en un grafo acíclicos dirigido con raíz
- Cada nodo tendrá ya sea dos o cero arcos salientes
- Los nodos con cero arcos salientes son llamados *nodos hoja*
- Habrá un nodo hoja para cada trapezoide en el mapa

Estructura de datos para localización de puntos

- El resto de los nodos se denominan *nodos internos*, y se utilizan para orientar la búsqueda hacia las hojas
- Atención esto no es un árbol binario, porque puede haber subárboles compartidos
- En esta estructura hay dos tipos de nodos internos: nodos x y nodos y
- Cada nodo x contiene la coordenada x_0 de un punto extremo de uno de los segmentos en S_i
- Sus dos hijos corresponden a los puntos que se extiende a la izquierda y a la derecha de la línea vertical $x = x_0$

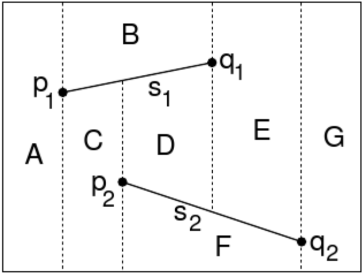
Estructura de datos para localización de puntos

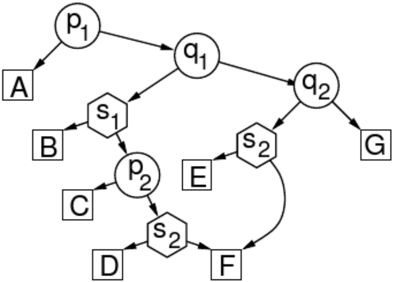
- Cada nodo y contiene un apuntador a un segmento de línea de la subdivisión planar
- Sus hijos izquierdo y derecho corresponden a si el punto de consulta está por encima o por debajo de la línea que contiene este segmento, respectivamente
- No se deje engañar por el nombre de nodo y, las comparaciones dependen tanto del valor x como del y del punto de consulta
- Tenga en cuenta que la búsqueda llegará a un nodo y sólo si ya hemos comprobado que la coordenada x del punto de consulta se encuentra dentro de la "placa" vertical que contiene este segmento

Estructura de datos para localización de puntos

- Nuestra construcción de la estructura de datos para localización de puntos refleja la construcción incremental del mapa trapezoidal
- En particular, si congelamos la construcción justo después de la inserción de cualquier segmento, la estructura actual será una estructura para localización de puntos para el mapa trapezoidal actual
- En la figura siguiente se muestra un ejemplo simple de como luce la estructura de datos para dos segmentos de línea
- Los nodos circulares son los nodos x y los hexagonales son nodos y
- Hay una hoja para cada trapezoide

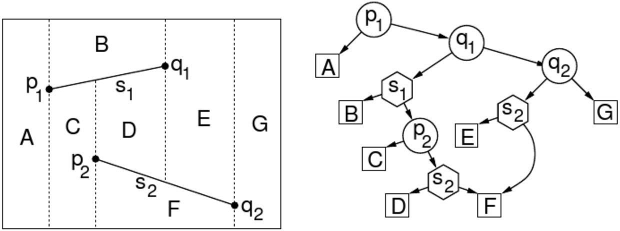
Estructura de datos para localización de puntos





Estructura de datos para localización de puntos

- Por ejemplo, si el punto de consulta está en el trapezoide D, detectaríamos primero que está a la derecha del punto extremo p_1 (hijo derecho), luego a la izquierda de q_1 (hijo izq.), entonces bajo s_1 (hijo derecho), luego a la derecha de p_2 (hijo derecho) y a continuación, por encima de s_2 (hijo izq).



1. Localización de puntos en el plano
   - Introducción
   - Mapas trapezoidales
   - Construcción de mapas trapezoidales
   - Análisis de complejidad
   - Estructura de datos para localización de puntos
   - **Construcción incremental**
   - Intersección de segmentos de línea (algoritmo aleatorizado)

Construcción incremental

- La pregunta es ¿cómo podemos construir esta estructura de datos de manera incremental?
- En primer lugar observemos que cuando un nuevo segmento de línea es agregado, sólo tenemos que ajustar la parte del grafo que implica los trapezoides que se han eliminado como resultado de esta nueva adición
- Cada trapezoide que se suprime se sustituye por una estructura de búsqueda que determina el recién creado trapezoide que lo contiene

Construcción incremental

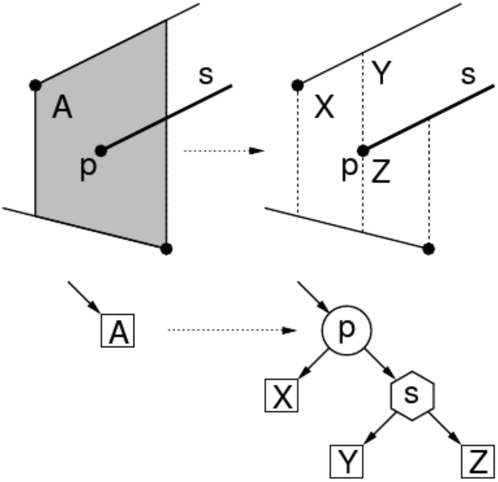
- Supongamos que añadimos un segmento de línea s
- Esto da lugar a la sustitución de un conjunto existente de trapezoides con un conjunto de nuevos trapezoides
- Como consecuencia de ello, vamos a sustituir las hojas, asociadas con cada uno de esos trapezoides suprimidos, con una pequeña estructura de búsqueda, la cual localiza el nuevo trapezoide que contiene el punto de consulta
- Hay tres casos que se plantean, dependiendo del número de puntos extremos de segmentos que caen dentro del trapezoide actual

Construcción incremental

Un solo punto extremo (izq. o derecho):

- Un solo trapezoide A se sustituirá por tres trapezoides, denotados X, Y y Z
- Sea p el punto extremo, creamos un nodo x para p, y un hijo es un nodo hoja para el trapezoide X que se encuentra fuera de la proyección vertical del segmento
- Para el otro hijo, creamos un nodo y cuyos hijos son los trapezoides Y y Z que caen por encima y por debajo del segmento, respectivamente

Construcción incremental



Construcción incremental

Ningún punto extremo:

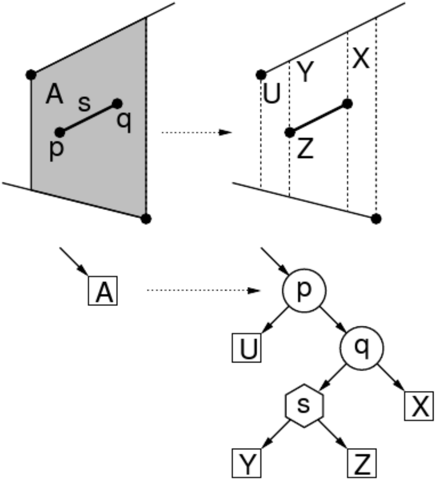
- Esto ocurre cuando el segmento corta y atraviesa completamente un trapezoide
- En este caso un trapezoide A se sustituirá por dos trapezoides, uno arriba y uno abajo del segmento, denotados Y y Z
- Sustituimos el nodo hoja para el trapezoidal original con un nodo y cuyos hijos son nodos hoja asociado con Y y Z

Construcción incremental

Dos puntos extremos:

- Esto ocurre cuando el segmento cae completamente dentro de un trapezoide
- En este caso, un trapezoide A se sustituirá por cuatro trapezoides, U, X, Y y Z
- Sean p y q los extremos izquierdo y derecho del segmento, creamos dos nodos x, uno para p y otro para q
- También se crea un nodo y para el segmento de línea, y se enlaza todo como se muestra en la siguiente figura

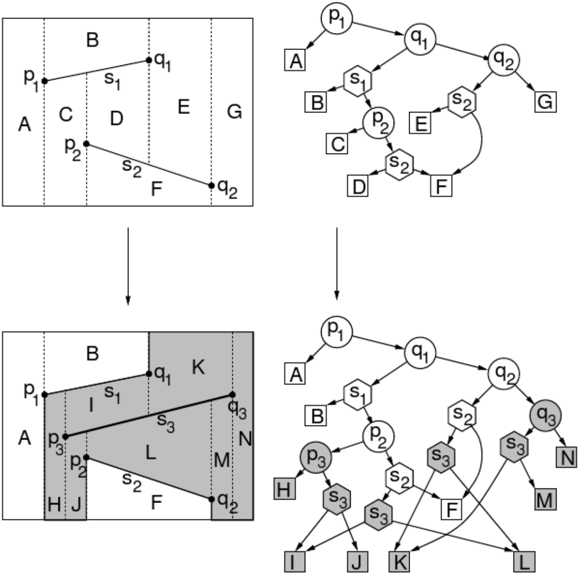
Construcción incremental



Construcción incremental

- Es importante señalar que (a través de compartir) cada trapezoide aparece exactamente una vez como una hoja en la estructura resultante
- Un ejemplo mostrando la transformación completa de la estructura de datos después de agregar un segmento se muestra en la siguiente figura

Construcción incremental



Construcción incremental

- Un ejercicio interesante consiste en realizar los siguientes análisis de complejidad:
  - Complejidad espacial de la estructura de datos para localización de puntos vista en clase
  - Complejidad temporal de una búsqueda en dicha estructura de datos

1. Localización de puntos en el plano
   - Introducción
   - Mapas trapezoidales
   - Construcción de mapas trapezoidales
   - Análisis de complejidad
   - Estructura de datos para localización de puntos
   - Construcción incremental
   - Intersección de segmentos de línea (algoritmo aleatorizado)

Intersección de segmentos de línea (algoritmo aleatorizado)

- En una clase anterior presentamos un algoritmo de barrido del plano para calcular la intersección de segmentos de línea que se ejecutaba en tiempo $O((n+I)\log n)$, donde I es el número de puntos de intersección

- Es importante notar que el enfoque aleatorizado que presentamos hoy puede ser adaptado para resolver también la intersección de segmentos de línea (ejercicio interesante)

- Dicha adaptación permite resolver el problema general de intersección de segmentos en tiempo $O(I + n\log n)$, con lo cual se remueve el factor log del término I