Un enfoque propuesto para las Búsquedas por Rangos con Separabilidad Geométrica

Edilma Olinda Gagliardi

LIDIC[1]
Departamento de Informática
Facultad de Ciencias Físico, Matemáticas y Naturales
Universidad Nacional de San Luis, Argentina
oli@unsl.edu.ar
Fax: 54-2652-430224

Gregorio Hernández Peñalver

Departamento de Matemática Aplicada
Facultad de Informática
Universidad Politécnica de Madrid, España
gregorio@fi.upm.es
Fax: 34-91-3367426

Resumen

Un problema que se presenta a menudo en Bases de datos es el estudio de los rangos y las consultas por rangos, denominado *Búsquedas por Rangos*. Este problema tratado desde una perspectiva geométrica nos permite diseñar y analizar los algoritmos y estructuras de datos utilizadas con herramientas propias de la Geometría Computacional.

En este trabajo presentamos una introducción a la temática, relacionándola específicamente a otra línea de investigación vigente de la Geometría: *Separabilidad Geométrica*.

El objetivo de esta propuesta es presentar los aspectos teóricos y prácticos relevantes para las búsquedas por rangos y separabilidad de objetos geométricos, realizando una vinculación entre ambas. Proponemos nuevas formas de obtención de esquemas de partición y estructuras adecuadas para la resolución de consultas por rangos.

Palabras Claves:

Búsquedas por rangos, Separabilidad geométrica, Partición simplicial, Cutting.

[1] Laboratorio de Investigación y Desarrollo en Inteligencia Computacional. Director: Dr. Raúl Gallard.

Este artículo es parcialmente subvencionado por el Proyecto de la UPM de Al2002-1010-2.43 Geometría Computacional, de la Universidad Politécnica de Madrid, España.

1. Introducción

La Geometría Computacional se interesa por demostrar la existencia de la solución de un problema y por encontrar los algoritmos y estructuras de datos eficientes, medidos respecto de su complejidad temporal y espacial respectivamente [AHU74]. Por tanto, esta disciplina forma parte de la teoría del diseño y análisis de algoritmos y estructuras de datos [Abe00], [BKOS97], [BY98], [GO97], [SU00], [Tou85], [Tou92]. Una línea de investigación vigente y con resultados recientes es la *Separabilidad de objetos geométricos* [Sea02]. Los criterios de separabilidad tienen aplicaciones interesantes, como por ejemplo el análisis de imágenes, la clasificación de datos, etc. Siempre que sea necesario discriminar objetos en un espacio de trabajo, por algún atributo del mismo, los criterios de separabilidad juegan un papel importante.

Nuestra introducción al estudio de la disciplina se basa en la relación existente con las bases de datos: un problema que se presenta a menudo es el estudio de los rangos y las consultas por rangos, denominado *Búsqueda por Rangos*. Esta temática vista desde una perspectiva geométrica, nos permite diseñar y analizar los algoritmos y estructuras de datos utilizados con instrumentos propios de la Geometría Computacional [AE98], [Aga97], [AM94], [BF79], [Mat94].

Nuestra propuesta consiste en vincular las Búsquedas por rangos con la Separabilidad geométrica, dado que el grado de avance en esta última temática ha sido relevante recientemente. En este sentido, las regiones se determinan en base a atributos propios de los objetos geométricos y de su ubicación en el espacio. Las consultas, básicamente, se reducen a casos particulares de las búsquedas por rangos.

En el presente trabajo, primeramente, introducimos las Búsquedas por rangos, las nociones de Separabilidad y la vinculación propuesta entre ambas temáticas. Por último, brindamos nuestras propuestas, conclusiones y perspectivas de trabajos futuros.

2. Búsquedas por Rangos

La Búsqueda por rangos es uno de los problemas centrales en Geometría Computacional, no sólo por la variedad de aplicaciones que posee sino porque, además, una gran cantidad de problemas geométricos pueden resolverse observándolos como problemas relacionados a las búsquedas por rango. Comenzamos con las nociones básicas.

Un *Espacio de Rangos* es un sistema $\Omega = (U, F)$, donde U es un conjunto de objetos geométricos y F es una familia de subconjuntos de U. Los elementos de F se llaman *Rangos* de Ω. El sistema Ω se llama *Espacio de Rango Finito* si el conjunto U es finito. Por ejemplo:

$\Omega_1 = (R^d, \{ h/\ h \text{ es un semiespacio en } R^d \})$, $\Omega_2 = (R^d, \{ h/\ h \text{ es una bola en } R^d \})$

Un problema de búsqueda por rangos puede expresarse del siguiente modo: Dados un espacio de rangos $\Omega = (U, F)$, un subconjunto S de objetos de U y un rango G de F, consultar los objetos geométricos que están en $S \cap G$. En este caso, a G se lo llama *rango de consulta* (*query range*).

Para resolver este problema de búsqueda por rangos, es necesario diseñar un algoritmo eficiente que resuelva tal intersección. El problema real para el cual tiene sentido este estudio, es cuando previamente se da el conjunto de objetos geométricos S y posteriormente se realizan repetidas consultas por la intersección, variando el rango G. En estos casos es donde se requiere diseñar una estructura de datos apropiada para almacenar el conjunto S, tal que frente a cada consulta por rango G, ésta se pueda responder eficientemente.

La Búsqueda por rangos, esencialmente, consiste en buscar los objetos geométricos que contiene una determinada región del espacio de objetos geométricos. Observando la figura o forma de las regiones correspondientes a los rangos de consultas, podemos hacer una clasificación en cuatro

tipos de búsqueda por rangos: *Búsqueda por Rangos Ortogonales* (*BRO*), *Búsqueda por Rangos SemiAlgebraicos* (*BRSa*), *Búsqueda por Rangos SemiEspaciales* (*BRSp*) y *Búsqueda por Rangos Simpliciales* (*BRSx*).

En cuanto a las consultas, dado un rango de consulta puede interesarnos efectuar alguna de las siguientes clases sobre él: *Consultas de Recuento* (*range counting*), *Consultas de Reporte* (*range reporting*), *Consultas Booleanas* (*range emptiness*) y *Consultas Especiales*.

La mayoría de las estructuras de datos para consultas por rango, se construyen en forma recursiva, dividiendo el espacio de objetos geométricos en varias regiones, con propiedades geométricas deseables sobre ellas. Estas estructuras de datos son referidas como esquemas de descomposición jerárquicos. Los análisis de las complejidades tanto en tiempo de construcción, en espacio de almacenamiento y en tiempos de respuestas, dependen de la dimensión del espacio, de la cardinalidad de *S*, de la estrategia de particionamiento de *S* y del tamaño de la respuesta de la consulta.

A continuación, nosotros describiremos la *Búsquedas por Rangos Simpliciales*, que son las de interés en este trabajo. Las consultas en este contexto son consultas en polígonos; es decir, los rangos son polígonos. Todo polígono puede descomponerse en piezas más pequeñas, realizando una triangulación del mismo. De esta forma, la consulta en el polígono se transforma en consultas sobre triángulos de la triangulación[2]. El resultado de la consulta es la unión de las respuestas de cada triángulo. Este concepto puede extenderse al espacio *d*-dimensional, donde tenemos símplices[3], [Mat91], [Mat92a], [Mat92c], [Sar98], [Wel88], [Wil82]. Una consulta por rango simplicial se reduce a tres búsquedas en semiespacios; por lo tanto, el problema se reduce a tratar básicamente consultas por semiespacios.

Supongamos que dado un conjunto *S* de puntos en el plano, queremos informar cuáles son los puntos de *S* que están contenidos en un cierto triángulo, el cual representa un rango de consulta. Dado que el tamaño de cualquier estructura y el tiempo de consulta tiene al menos complejidad lineal y logarítmica respectivamente, consideremos estos extremos: i) ¿Cuán rápido se puede contestar una consulta de rango simplicial usando una estructura de tamaño lineal? y ii) ¿Qué espacio se requiere para una estructura de datos que conteste una consulta en tiempo logarítmico? La combinación de estos dos extremos nos lleva a una competencia espacio-tiempo. Nosotros dividiremos nuestra exposición en dos partes, atendiendo a estos dos interrogantes.

2.1 Algoritmos que utilizan espacio lineal (o casi lineal)

La mayoría de las estructuras de datos de tamaño orden lineal se basan en los *Árboles de Partición* (*Partition Trees*).

Dado un conjunto *S* de cardinalidad es *n*, una *Partición Simplicial* de *S* es un conjunto de la forma $\Pi = \{(S_1,\Delta_1),...,(S_r,\Delta_r)\}$, que cumple las siguientes condiciones: S_i es una *clase* y Δ_i es un símplice, $S_i \subseteq S$, $S_i \subseteq \Delta_i$, $S_i \cap S_j = \phi$ para $i \neq j$. El *número de corte* (*crossing number*) *de una recta l con respecto a* Π es el número de triángulos de Π cortados por *l*. El *número de corte de* Π es el máximo número de cortes sobre todas las posibles rectas *l*. La partición simplicial es *fina* si $|S_i| \leq 2n/r$, para cada $1 \leq i \leq r$.

Un *Árbol de Partición* (*Partition Tree*) para *S* está formado así: la raíz tiene *r* hijos, siendo cada uno la raíz de un árbol de partición definido recursivamente para cada una de las clases de la partición simplicial fina. La estructura básica es la siguiente: Si *S* contiene un único punto *p*, el árbol de partición consiste de una hoja, donde *p* se almacena explícitamente. En otro caso, la

[2] Es lo que se denomina *Rango simplicial.*
[3] Un símplice es el cierre convexo de *d*+1 puntos independientes.

estructura es un árbol T con r hijos. Cada hijo de la raíz corresponde a uno de los triángulos de la partición de S. El triángulo de la partición correspondiente a un hijo v se denota t_v. La correspondiente clase S_v se llama clase *canónica* de v. El hijo v es la raíz del árbol de partición T_v definido recursivamente sobre el conjunto S_v. Con cada hijo v, almacenamos el triángulo t_v con información acerca del subconjunto S_v, tal como su cardinalidad u otro dato de interés.

Podemos responder una consulta por un semiplano H. Para ello, basta con considerar los puntos de las regiones no atravesadas por la recta h que define H. Puede ocurrir que, o bien, toda la región esté contenida en el semiplano, en cuyo caso la región completa es parte de la respuesta; o que ninguna región esté contenida en H, en cuyo caso pierde su consideración futura.

Esto nos brinda un poco de eficiencia en cuanto a que sólo resta tratar aquellas regiones atravesadas por la recta borde h del semiplano de consulta H, del mismo modo. Aquí es donde juega un rol importante el número de cruce de la partición simplicial. El tiempo de respuesta de una consulta por semiplano está acotado a $O\left(\sqrt{n}\right)$ [AE98].

Volviendo a las consultas por rango simplicial, para responderlas podemos utilizar la misma estructura y el mismo algoritmo que para la consulta por semiplano, que sólo varía levemente el tiempo de respuesta. Un triángulo de la partición simplicial puede ser cortado por una recta borde del rango simplicial, la cual puede a lo sumo cortar $c\sqrt{r}$ triángulos de la partición. Como el rango simplicial define a lo sumo tres rectas, entonces en total cruza a lo sumo $3c\sqrt{r}$ triángulos. Por lo tanto, sólo hemos cambiado de c a $3c$, permaneciendo el tiempo de consulta cercano al visto, aunque que asintóticamente permanece en el mismo orden.

Esta estructura utiliza $O(n)$ espacio de almacenamiento; los puntos de S contenidos en un triángulo de consulta pueden ser contados en $O(n^{1/2+\varepsilon})$ tiempo y los puntos pueden ser reportados en $O(k)$ tiempo adicional, donde k es el número de puntos informados. La estructura puede ser construida en $O\left(n^{1+\varepsilon}\right)$.

2.2 Algoritmos con tiempo de respuesta logarítmico

Podemos mejorar los tiempos de consulta cambiando de $O\left(\sqrt{n}\right)$ a $O(\log n)$. La idea subyacente es la misma que en los árboles de partición con particiones simpliciales disjuntas, sólo que nos trasladamos al espacio dual.

Sea $L=\{l_1, l_2, ..., l_n\}$ un conjunto de rectas obtenidos después de dualizar[4] una nube S de puntos; sea r un parámetro, $1 \le r \le n$; se dice que una recta corta un triángulo si interseca su interior. Se dice que un $(1/r)$-*cutting* para L es un conjunto $C(L)=\{t_1, t_2, ..., t_m\}$ de triángulos posiblemente ilimitados con interiores disjuntos que cubren el plano, con la propiedad de que ningún triángulo de la partición es cortado por más de n/r rectas de L. El *tamaño* del cutting $C(L)$ está definido por la cantidad de triángulos que posee.

La estructura de datos basada en un cutting se llama *Árbol de Cortes* (*Cutting Tree*). Básicamente, se construye del siguiente modo: Si la cardinalidad de L es uno, entonces el árbol de cortes consiste de un único nodo en el que L se almacena explícitamente y él es el conjunto canónico del nodo. Si la cardinalidad es mayor que uno, tenemos un árbol T; entonces existe una correspondencia biunívoca entre los hijos de la raíz del árbol y los triángulos de un $(1/r)$-cutting $C(L)$ para el conjunto L, donde r se elige como una constante suficientemente grande. Cada

[4] Un punto en el plano está definido por su coordenada X y su coordenada Y. Una recta en el plano también está definida por dos parámetros, su pendiente y su corte con el eje Y. Podemos definir una aplicación biunívoca entre estos dos conjuntos de elementos de la siguiente manera: $p(p_x, p_y) \mid \rightarrow p^* \equiv r$: $(y=p_x x - p_y)$; r: $y=m_x+n \mid \rightarrow r^* \equiv p(m.-n)$.

triángulo del cutting se corresponde con un hijo v y se denota $t(v)$. El subconjunto de rectas de L que pasan por debajo[5] de $t(v)$ se llama *subconjunto canónico inferior de v* y es denotado $L^-(v)$. El subconjunto de rectas de L que pasan por encima de $t(v)$ se llama *subconjunto canónico superior de v* y es denotado $L^+(v)$. El subconjunto de rectas de L que cortan $t(v)$ se llama *subconjunto de cortes de t(v)*. El nodo v es la raíz de un árbol de cortes definido recursivamente sobre su conjunto de cortes y se denota Tv. En cada nodo v, almacenamos el triángulo $t(v)$, los subconjuntos $L^-(v)$ y $L^+(v)$. Para contar el número de rectas que pasan por debajo del punto de consulta consideramos la cardinalidad de $L^-(v)$; respectivamente para contar el número de rectas que pasan por encima del punto de consulta consideramos $L^+(v)$.

Para la consulta por rango simplicial, se tiene que dado un conjunto S de puntos en el plano se requiere contar el número de puntos contenidos en un triángulo de consulta. Para resolverlo, seguimos el enfoque de consultas por semiplanos, pero nos mudamos al plano dual. Así, el conjunto de puntos se dualiza en un conjunto de rectas. En el plano primal, un triángulo es la intersección de tres semiplanos, por lo que en el plano dual se dualiza en tres puntos. En consecuencia, en el plano primal decimos que el punto q está contenido en el triángulo sí y sólo sí está contenido en cada uno de los semiplanos. El enunciado dual de la búsqueda por rango triangular se plantea del siguiente modo: dado un conjunto L de rectas en el plano y tres puntos q_1, q_2 y q_3 de consulta rotulados como "superior" o "inferior", se requiere contar el número de rectas de L contenidas en los lados especificados de los tres puntos de consulta (en el plano dual). Este problema puede ser resuelto con un árbol de cortes de tres niveles.

Dado L un conjunto de n rectas en el plano, al usar un árbol de cortes, las rectas de L que pasan por debajo de un punto q de consulta pueden ser seleccionadas en $O(\log n)$ tiempo, con $O(\log n)$ subconjuntos canónicos. Por lo tanto, el número de tales rectas se puede contar en $O(\log n)$ tiempo. Para cualquier $\varepsilon>0$, un árbol de cortes sobre L puede ser construido utilizando $O(n^{2+\varepsilon})$espacio.

Las búsquedas por rangos simpliciales han recibido mayor atención recientemente, puesto que además de sus aplicaciones directas, han brindado soluciones a problemas subyacentes en problemas geométricos de mayor jerarquía. Con esto concluimos dos enfoques para su resolución.

3. Separabilidad Geométrica

Se dice que un conjunto finito S de superficies es un *separador* de conjuntos de objetos en R^d si las componentes conexas de R^d-S contienen objetos de un solo conjunto.

Los separadores geométricos son de utilidad en campos de aplicación donde se requiere discriminar o separar objetos. Nosotros presentamos diversos criterios de separabilidad de objetos geométricos. Los trabajos de separabilidad están orientados a dos o más conjuntos disjuntos de objetos geométricos, básicamente para puntos en el plano.

Algunas aplicaciones en las que la Separabilidad está inmersa son la separación de objetos de sus moldes; la separación de conjuntos bajo diferentes clases de movimientos y diversas definiciones de separabilidad [Tou85b]; la separación de dos polígonos simples por una traslación de uno de ellos tal que no exista colisión [Tou89], [TE84], [AS97]; el problema de *shattering* (*hacer pedazos*) [Fre00], [FMP90], [Nan99], [ES97]; etc.

[5] En términos generales, para decidir la ubicación de un punto p respecto de una recta l, seguimos el sentido contrario de las agujas del reloj y considerando los puntos q, z y p, y obtenemos el signo del cálculo del área signada. Si el valor del área es positivo, se concluye que el punto p está encima de la recta l; si es negativo, entonces está por debajo de la recta l. En caso de valor cero, los tres puntos son colineales.

Otro problema se da cuando tenemos dos conjuntos disjuntos de objetos en el espacio euclideano y nos preguntamos si es posible generar una superficie que separe ambos conjuntos. Esta pregunta se puede extender al espacio y a diversas clases de objetos.

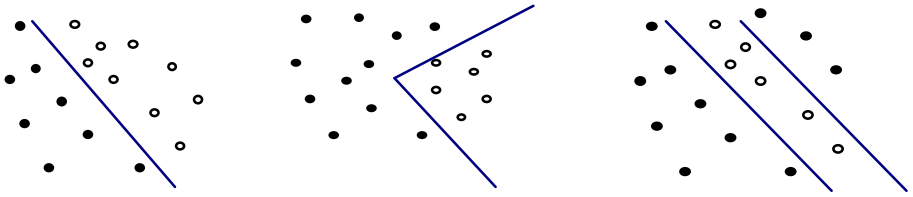
Para modelar este problema, se dan dos conjuntos disjuntos *P* y *Q* de objetos en el plano, a los que nos referiremos objetos *rojos* y *azules*, respectivamente. Eventualmente, los objetos pueden ser: puntos, segmentos, polígonos o círculos. En el caso de los polígonos, *n* y *m* representan el número total de segmentos de los polígonos de *P* y de *Q*. En otro caso, *n* y *m* representan la cardinalidad de *P* y *Q*, respectivamente. En cualquiera de los casos, *N* es el máximo de *n* y *m*. Entonces, dada una familia *C* de curvas en el plano, decimos que los conjuntos *P* y *Q* son *C-separables* si existe una curva $C_i \in C$, llamada *separador*, tal que cada componente conexa de $R^2 - C_i$ contiene solamente objetos, o bien de *P*, o bien de *Q*.

Describimos brevemente a continuación diferentes clases de separabilidad, dando las cotas de complejidad de los algoritmos correspondientes. No presentamos ninguna descripción de su funcionamiento, dado que no es el punto central, sólo nos interesa saber que existen. Para estudiar el tema en detalle, ver [Sea02].

Si *C* es la familia de rectas en el plano, tenemos la *separabilidad lineal*. Dos conjuntos *P* y *Q* son *linealmente separables* si y sólo si sus cierres convexos no se intersecan.

Dos conjuntos disjuntos *P* y *Q* de objetos en el plano son *separables por una cuña* si existe una cuña que contiene solamente todos los objetos de uno de los conjuntos. Aquí se estudia el problema de decidir la separabilidad por cuñas calculando la ubicación de los posibles ápices de cuñas; como una extensión al problema, se estudia hallar la cuña de ángulo mínimo (máximo).

Dos conjuntos disjuntos *P* y *Q* de objetos en el plano son *separables por banda* si existe una banda, determinada por dos rectas paralelas, que contiene solamente todos los objetos de uno de los conjuntos. Aquí se estudia el problema de decidir la separabilidad por bandas calculando el conjunto de intervalos de pendientes de bandas separadoras; como una extensión al problema, se estudia hallar la banda mínima (máxima).



Separabilidad líneal Separabilidad por Cuña Separabilidad por Banda

En este contexto, se plantean dos tipos de problemas:

i) *Problemas de Decisión o Existencia*: Dada la nube de puntos $P \cup Q$, de puntos rojos y azules en el plano, se cuestiona la existencia de un separador de los puntos rojos de los azules. Por ejemplo *¿existe una cuña que separa los puntos rojos de los azules?* En caso afirmativo, hallar los vértices de todas las cuñas separadoras.

ii) *Problemas de Optimización*: Asociado a la separabilidad por cuñas y bandas consisten en hallar la cuña separadora de mínimo (máximo) ángulo y la banda de mínimo (máximo) ancho que separan los puntos rojos de los azules.

Los algoritmos para resolver los problemas de decisión y optimización propuestos anteriormente se ejecutan en $O(N \log N)$ tiempo; salvo en el caso de la separación lineal, que toma $O(N)$ tiempo [Sea00].

Los estudios sobre separabilidad en el plano han avanzado sobre separaciones con varias bandas, cuñas y sectores. Así tenemos *separabilidad por dos bandas*, asociado al problema de optimización

de hallar las dos bandas de ancho mínimo (máximo). Este problema puede ser resuelto en $O(n^3 \log n)$ tiempo. En el mismo tiempo se puede decidir si las bandas son de mínima (máxima) anchura. En cambio, si se proporciona la pendiente de una de las bandas, el tiempo se reduce a $O(n^2 \log n)$, pudiendo hallarse las dos bandas de mínima (máxima) anchura en el mismo tiempo. Aun mejor si las dos pendientes son conocidas, puesto que se reduce a $O(n)$ tiempo. Como resultado, tenemos que dados dos conjuntos disjuntos P y Q de puntos en el plano, el mínimo número de bandas paralelas (rectas) que los separan y los correspondientes intervalos de pendientes, puede ser determinado en $O(n^2 \log n)$ tiempo.

También, tenemos *separabilidad por dos cuñas*, que en caso de existir el par de cuñas, podemos tener cuñas disjuntos o no; podemos determinarla en $O(n^5 \log n)$ tiempo; pero, en caso de que las cuñas sean disjuntas, podemos decidir la separabilidad en $O(n^3 \log n)$tiempo. Hay más variantes asociadas, como la *separabilidad por sectores*, si todas las cuñas separadoras tienen el mismo ápice.

En lugar de tener una nube bicolor, podemos considerar diversos colores. Dados $C_1, C_2, ..., C_k$ conjuntos disjuntos en el plano, decimos que el conjunto C_i tiene el color c_i, denotamos con n_i la cardinalidad de C_i y $n_1 + n_2 + ... + n_k = n$. Entonces, un conjunto finito S de curvas en el plano es un separador para los conjuntos $C_1, C_2, ..., C_k$ si cada componente conexa en $R^2 - S$ contiene elementos provenientes de uno solo de los conjuntos C_i. También decimos que cada componente es *monocromática*. Obviamente, cuando k=2, son los casos estudiados previamente.

Dados los conjuntos disjuntos $C_1, C_2, ..., C_k$ coloreados, con n puntos en total, en el plano, la mínima cantidad de rectas paralelas separadoras de los puntos en bandas monocromáticas y los intervalos de pendientes de todas las posibles soluciones, pueden calcularse en $O(n^2 \log n)$ tiempo. En el caso de permitir k-1 rectas paralelas únicamente, conseguir los intervalos de pendientes de todas las posibles soluciones, pueden calcularse en $O(kn) + O(k^2 \log k)$ tiempo.

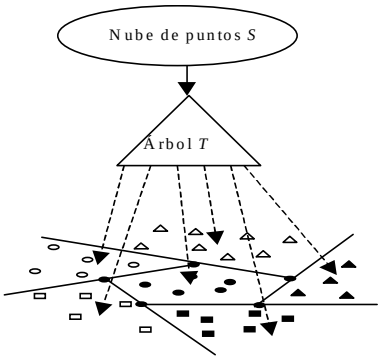
Los principales resultados sobre separabilidad con bandas y cuñas, con diferentes versiones sobre combinaciones y propiedades de ellas, son algoritmos eficientes para decidir la existencia de los separadores y calcular soluciones factibles. Algunos de estos resultados pueden ser extendidos en el plano al uso de otros objetos geométricos que no sean puntos y al espacio tridimensional en donde sólo existen algunos algoritmos para resolver algunos de los problemas vistos.

En este marco, también queda abierto el estudio sobre nuevos criterios de separabilidad; es decir, no sólo usar planos o semiplanos, sino otros objetos como pirámides dobles (bipirámides), cilindros, cónicas, cónicas dobles, etc.

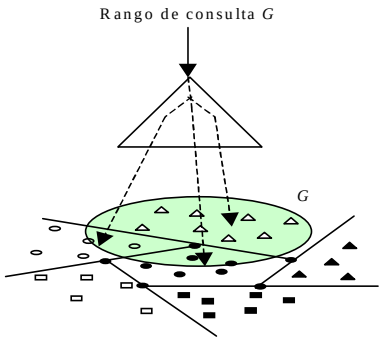
4. Un enfoque propuesto con Separabilidad Geométrica

En esta sección queremos unificar las nociones de Búsquedas por rangos con las de Separabilidad geométrica.

Como hemos visto acerca de Separabilidad geométrica, tenemos conjuntos disjuntos de objetos en el plano, los cuales queremos separar acorde a algún criterio de separabilidad. Los conjuntos son disjuntos debido a alguna propiedad específica de los objetos y de su posición en el espacio. Nos interesan particularmente las descripciones de las curvas que determinan las regiones que contienen tales conjuntos disjuntos, dado que ellas constituyen los separadores geométricos. Con esta información podemos obtener un esquema de partición, en cuyo caso estamos en condiciones de crear una estructura de datos apropiada para acceder a los puntos. En la siguiente figura, podemos visualizar estas ideas:



Nube de puntos, dividida en regiones por alguna característica de los mismos



Regiones seleccionadas para la búsqueda

Con respecto a las consultas, podemos realizar la búsqueda por las características dadas de los puntos, o bien, por rangos. En el primer caso, por ejemplo, podemos solicitar las regiones correspondientes a los puntos rojos, pudiendo obtener como respuesta una o varias regiones. O bien, consultar cuáles son los diferentes colores existentes. También, podemos utilizar un rango de consulta, que puede abarcar parte de una región o varias regiones. Por ejemplo, un rango de consulta dado por un semiplano *H*, definido por una recta *h*, puede corresponderse con partes de regiones, con regiones completas, y posiblemente (es deseable), dejar excluidas regiones íntegras.

A continuación veremos la construcción del primer nivel de una estructura jerárquica, considerando una partición obtenida por medio de separabilidad geométrica, y luego analizaremos la resolución de consultas.

Cualquier característica que interese acerca de los puntos, puede verse como una aplicación $w:S \rightarrow C$, donde $C=\{c_1, c_2,..., c_k\}$, $1 \leq i \leq k$, para todo p perteneciente a S.

Dado un conjunto S de n puntos en el plano y una función total $w:S \rightarrow C$, con $C=\{c_1, c_2,..., c_k\}$, generamos una *Partición Poligonal* de S, $\Pi=\{(S_1,P_1),(S_2,P_2),...,(S_m,P_m)\}$, con las siguientes propiedades:

- Cada es un subconjunto de S, (S_i es una clase), cada P_i es una región poligonal y para cada i existe un h tal que $w(p) = c_h$, si p está en S_i.
- $S_i \subseteq S$, $S_i \subseteq P_i$, $S_i \cap S_j = \emptyset$, $P_i \cap P_j = \emptyset$ para $i \neq j$, $1 \leq i,j \leq m$.
- Para cada i existe un h tal que si p está en P_i , $w(p)=c_h$.
- n_i es la cardinalidad de S_i y $n_1 + n_2 + ... + n_m = n$.
- Las regiones poligonales son inducidas por separadores geométricos.

Los criterios de separabilidad utilizados para obtener las regiones poligonales pueden ser: lineal, por cuñas, por bandas, por doble cuña, etc. Es posible utilizar herramientas inteligentes[6] que nos brinden información acerca de cuál criterio de separabilidad es el más apropiado.

Es interesante y deseable pensar que las nubes de puntos acepten criterios de separabilidad acotados a complejidades bajas, aunque no es relevante, porque la principal presunción que podemos hacer es que la estructura se genera una única vez. Como hemos visto, puede ser prioritario crear estructuras con bajo costo de almacenamiento o bien crear estructuras que privilegien los tiempos de respuestas. Por asumir, justamente, que la estructura se crea una vez y que es estática, es que los costos de construcción son amortizados en base al criterio de ahorro

[6] Algunas herramientas de Inteligencia Computacional incluyen Metaheurísticas, Redes Neuronales, Algoritmos Evolutivos, Inteligencia Artificial Distribuida, Agentes, etc.

elegido. Además, el objetivo principal es realizar muchas consultas por diferentes regiones sobre la misma nube de puntos.

En comienzo, elegimos trabajar con particiones poligonales y árboles de partición. A continuación describimos el árbol de partición correspondiente al primer nivel:

Teniendo una partición poligonal $\Pi=\{(S_1,P_1),(S_2,P_2),...,(S_m,P_m)\}$ para una nube de puntos S, un *Árbol de Partición Poligonal* en su primer nivel se construye del siguiente modo:

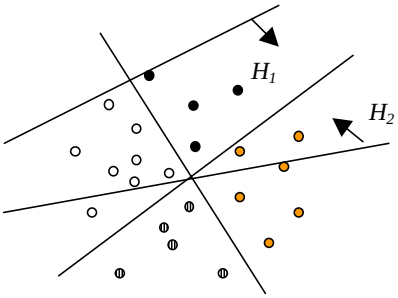
- Si S contiene un solo punto p, el árbol de partición consiste de una hoja donde p se almacena explícitamente. En este caso, la partición poligonal es $\Pi=\{(S_1,P_1)\}$.
- En otro caso, la estructura es un árbol T con m hijos. Cada hijo de la raíz corresponde a una de las regiones poligonales de la partición de S. La región poligonal correspondiente a un nodo v, hijo de la raíz, se denota P_v. La clase respectiva, S_v, se llama clase canónica de v.
- En cada hijo de la raíz almacenamos información correspondiente a la clase S_v y la descripción de la región poligonal P_v.

Con respecto a las consultas, tenemos básicamente del tipo: *¿cuáles son las regiones de un determinado color?*, o, *¿cuáles son los diferentes colores que se hallan en el plano?* O bien, las *consultas por rango*.

Para la primera, basta con consultar por el color correspondiente a cada región poligonal P_v, descripto en S_v. Es una consulta que toma $O(m)$ tiempo, siendo m la cantidad de regiones poligonales que obtuvimos. Para la segunda consulta, es la misma idea, salvo que por cada región poligonal, registramos qué color almacena y finalmente, se concluye en la unión de todos los colores existentes. También su complejidad es lineal.

Acerca de las consultas por rangos, supongamos un semiplano H, definido por una recta h. Si una región poligonal P_i no es cortada por la recta h, una de las siguientes situaciones ocurre: $P_i \subseteq H$ o bien $P_i \cap H = \emptyset$. En el primer caso, todo P_i es parte de la respuesta, mientras que en el segundo, P_i se descarta totalmente. Queda por ver el caso en que P_i es cortado por h, en cuyo caso basta revisar la clase S_i y determinar qué puntos quedan contenidos en el semiplano H. En este último caso, la búsqueda es lineal sobre S_i. Como sólo tenemos un nivel del árbol, hasta aquí en el peor caso, el orden de complejidad de la consulta es lineal.

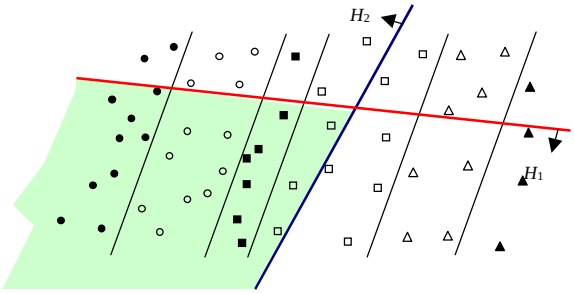
En la siguiente figura, mostramos dos semiplanos de consulta.



El semiplano H_1 tiene peores condiciones, puesto que por cada sector ocurre: $P_i \cap H \neq \emptyset$ y $P_i \not\subset H$, y por lo tanto es necesario determinar para todos los puntos de S en qué semiplano quedan contenidos.

Mientras que para H_2, solamente hay que mirar en detalle los dos sectores que corta. El resto o queda fuera o queda totalmente contenido en el semiplano de consulta.

Veamos otro ejemplo, supongamos que proponemos un *rango cuña*. Una cuña no es más la región correspondiente a la intersección de dos semiplanos opuestos; más precisamente es alguno de los cuadrantes descriptos por dos rectas que se cortan en algún punto. En la siguiente figura mostramos un rango cuña sombreado.



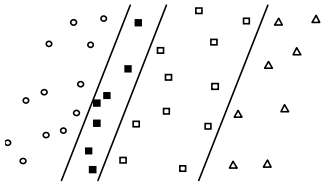
Una consulta de este tipo se descompone en dos consultas por semiplanos, y luego se realiza la intersección de los cuadrantes correspondientes.

Un *rango por bandas*, no difiere mucho en su resolución y complejidad.

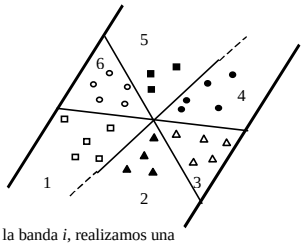
Consideremos a continuación cada región poligonal en particular. ¿Qué cosas podemos continuar haciendo? Se abre un espectro de propuestas. Cada región poligonal constituye una nueva nube de puntos sobre la cual trabajar. Esencialmente, hay tres enfoques vistos que podemos aplicar: Partición simplicial fina, Cutting y Separabilidad geométrica.

Cualquiera de ellos nos lleva construir estructuras de datos jerárquicas. Estos árboles quedarán pendientes de cada hijo de la raíz del árbol de partición poligonal. De este modo, el árbol de partición poligonal tiene su primer nivel formado por Separabilidad geométrica. A partir del segundo nivel, se forma según la alternativa elegida: o nuevamente un árbol de partición, o un árbol de cortes, o un árbol de partición poligonal. Veamos algunos ejemplos en donde seleccionamos la última alternativa.

Ejemplo 1[7]: Supongamos que realizamos una separación por varias bandas. Todas las regiones obtenidas son parcialmente acotadas. En el árbol que pretendemos construir, la nube completa queda representada en la raíz del árbol de partición y cada banda en un hijo de la raíz, dando origen a un árbol de aridad *m* en el primer nivel. En este caso, *m=k*. Luego, por cada hijo analizamos qué posibilidades de separación existen y en base a ello, volvemos a aplicar algunas de las vistas previamente. Por ejemplo, supongamos que tomamos la banda *i*, y que vale separarla en sectores.



En el primer nivel realizamos una Separación por varias bandas

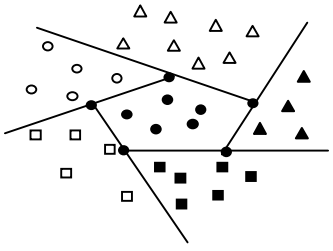


En la banda *i*, realizamos una Separación por sectores

Los sectores 2 y 5 son parcialmente acotados por las rectas que definen las cuñas. Mientras que, los sectores 1, 3, 4 y 6 están totalmente acotados por las rectas que definen la banda y las que definen las cuñas

Con respecto a la banda *i*, del *i*-ésimo hijo del nodo raíz se desprenderán seis nodos hijos, uno para cada sector obtenido por la segunda separación. Repitiendo este tipo de proceso con cada hijo, estamos en condiciones de conformar el segundo nivel del árbol.

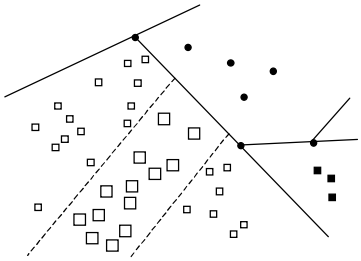
Ejemplo 2: Supongamos que tomamos la siguiente descomposición en primera instancia.



La nube *S* está descompuesta en seis regiones. Tenemos $k=6$ conjuntos disjuntos de puntos, coloreados diferentes. El ejemplo muestra un cierre convexo para el conjunto C_1. Por cada lado del cierre, se disparan rayos que parten la zona exterior en varias regiones que cubren el espacio. En cada región se alberga un C_i, con $2 \le i \le k$. También en este caso tenemos $m=k$.

[7] En las siguientes figuras, las formas de los puntos intentan representar diversos valores para la propiedad que se esté analizando; en ningún caso representan formas geométricas.

El primer nivel del árbol de partición poligonal queda compuesto por seis nodos pendientes de la raíz, cada uno representando una región poligonal. Elegimos una región cualquiera y trabajamos sobre ella. Si la región corresponde al *i*-ésimo hijo de la raíz, éste será el padre el tres nodos, correspondientes a cada banda obtenida en la segunda separación.



El separador es una banda, posiblemente de ancho máximo.

5. Propuestas

Hemos creado un árbol de partición en su primer nivel, obtenido por medio de Separabilidad geométrica. Nos planteamos qué cosas podíamos continuar haciendo, y vemos que se abre un abanico de posibilidades, pero que, básicamente, se pueden resumir en tres enfoques: *Partición simplicial*, *Cutting* y *Separabilidad geométrica*. En base a ello, a continuación resumimos las propuestas para el segundo nivel (y/o siguientes) del árbol de partición poligonal:

- Propuesta 1: *Crear una partición simplicial fina en cada región poligonal.*

Podemos tomar cada región poligonal y crear sobre ella una partición simplicial fina, a los fines de mejorar los tiempos de consultas que implican $P_i \cap H \neq \emptyset$ y $P_i \not\subset H$. En este caso, la nube de puntos está conformada por S_v y el espacio por P_v. Sobre ella creamos el árbol de partición simplicial correspondiente, y repetimos este proceso en cada nodo del primer nivel.

- Propuesta 2: *Crear un cutting por cada región poligonal*

Por cada región poligonal, creamos un cutting y el árbol de cortes correspondiente. En este caso, dualizamos la nube de puntos S_v y creamos en base a L_v (conjunto de rectas que dualizan los puntos de S_v) un $(1/r)$-cutting $C(L_v)$. Considerar que r se determina en función de n_v, que es la cardinalidad de S_v. En base al cutting, creamos el árbol de corte correspondiente y aplicamos las técnicas presentadas para las búsquedas.

- Propuesta 3: *Particionar por algún criterio de separabilidad*

Por cada hijo de la raíz podemos analizar qué posibilidades de separación existen y en base a ello, volvemos a aplicar algunas de las variantes vistas previamente. Repitiendo proceso con cada hijo, estamos en condiciones de conformar el segundo nivel del árbol. Observemos que la clasificación en este nivel debe realizarse en función de otra característica o atributo del punto.

Hemos mencionado que *podemos analizar qué posibilidades de separación existen*, sin aludir nociones al respecto; tal análisis no es elemental. Si de alguna manera, nosotros conocemos a priori cuál es el criterio de separabilidad más propicio para la nube de puntos, entonces los trabajos se simplifican. Una propuesta para este último punto, es trabajar con la aplicación de herramientas inteligentes, con lo cual se podrían lograr mejores resultados, puesto que se optimiza la selección del criterio de separación. Ésta es otra línea de trabajo que queda abierta a investigaciones y desarrollos futuros.

6. Conclusiones y visión de futuro

El grupo de trabajo en Geometría Computacional de la UNSL, con el asesoramiento de docentes de la UPM, han dado inicio a un proyecto de investigación conjunto, con el objetivo principal de

consolidar la línea de trabajo en la UNSL, aportando nuevos enfoques y técnicas algorítmicas a las líneas de investigación ya establecidas en su Departamento de Informática dentro de LIDIC. Este trabajo se enmarca dentro de tal proyecto.

En cuanto a integrar las nociones de Separabilidad geométrica y Búsquedas por rango, tenemos que las posibilidades de combinación son muchas. Algunos casos serán sencillos de analizar y de crear las particiones, estructuras y algoritmos correspondientes. Pero, seguramente, una gran cantidad requerirán de mucho análisis y trabajo futuro. Este trabajo es resultado de tesis de maestría en Ciencias de la Computación, de la UNSL.

Algunos de los resultados expuestos pueden ser extendidos en el plano al uso de otros objetos geométricos y al espacio tridimensional en donde sólo existen algunos algoritmos para resolver algunos de los problemas vistos. En este marco, también queda abierto el estudio sobre nuevos criterios de separabilidad, utilizando otros objetos como pirámides dobles (bipirámides), cilindros, cónicas, cónicas dobles, etc. Las ideas presentadas, pretenden llegar a árboles decisión de *k* niveles. Aunque aquí la problemática esté clasificada como problemas *NP-hard*.

Agradecimientos: Queremos agradecer a los árbitros anónimos que con sus comentarios ayudaron a mejorar esta presentación.

Referencias bibliográficas

[Abe00] Abellanas Oar, M. *Descubriendo la Geometría Algorítmica, 2000*. http://www.dma.fi.upm.es/mabellanas/divulgación/GeometriaAlgoritmica.html

[AE98] Agarwal, P.; Erickson, J. *Geometric range searching and its relatives*; Advances in Discrete and Comput. Geom. (B. Chazelle,J. Goodman, and R. Pollack, eds.), American Mathematical Society, Providence, 1998.

[Aga97] Agarwal, P.; *Range searching* CRC Handbook of Computational Geometry (J. Goodmand and J. O'Rourke, eds.).

[AHU74] Aho, A.V.; Hopcroft, J. E.; Ullman, J. *The desing and analysis of computer algorithms*, Addison-Wesley Series in Computer Science and Information Processing, 1974.

[AM94] Agarwal, P.; Matousek,J.; *On range searching with semialgebraic sets;* Discrete Comput. Geom., 11: 393-418, 1994.

[AS97] Aronov, B.; Sharir, M. *On traslational Motion Planing of a Convex Polyhedron in 3-space*, SIAM J. Computing, Vol.26, No.6, 1785-1803, 1997.

[BF79] Bentley, J.L.; Friedman, J.H; *Data Structures for range searching*; ACM Comput. Surv. 11:397-409;1979.

[BKOS97] de Berg, M; Kreveld, Overmars, M; Schwarzkopf. *Computational Geometry: algorithms and applications*, Springer Verlag, BH 1997

[BY98] Boissonnat, J.D.; Yvinec, M. *Algorithmic Geometry*, Cambridge University Press, 1998.

[ES97] Efrat, A.; Schwarzkopf, O.; *Separating ∧ Shattering long line segments*, Information Processing Letters, 64, 309-314, 1997.

[FMP90] Freimer, R.; Mitchell, J.S:B.; *On the Complexity of Shattering using Arrangementes* . Proc. Of 2nd. Canadian Conference on Computational Geometry, 218-222, 1990.

[Fre00] Freimer, R.; *Investigations in Geometric Subdivisions: Linear Shattering and Cartographic map Coloring*. Ph. D. Thesis, Cornell University, 2000.

[GO97] Goodman, J.; O'Rourke, J. *Handbook of Discrete and Computational Geometry*. CRC Press 1997.

[Mat91] Matousek, J. *Cutting hyperplane arrangements*; Discrette Comput. Geom., 6(5):385-406, 1991.

[Mat92a] Matousek, J. *Efficient partition tree*; Discrette Comput. Geom., 8:315-334, 1992.

[Mat92c] Matousek, J.*Range searching with efficient hierarchical cuttings*; Proc. 8 ACM Symposium on Computational Geometry, 276-285, 1992.

[Mat94] Matousek,J.; *Geometric Range searching*; ACM Comput. Survey, 26:421-461;1994.

[Meg79] Meggido, N.; *Combinatorial optimization with rational objective functions.*Math.Oper.Res.4:414-424;1979.

[Nan99] Nandy, S.C.; *Shattering a Set of Objects in 2D*, Proc. Of the 11th canadian Conference on Computational Geometry, 1999.

[Sar98] Sariel, Har-Peled; *Constructing planar cuttings in theory and practice*;

[Sea02] Seara, C; *On geometric separability*; Tesis Doctoral 2002, Univ. Politécnica de Cataluña.

[SU00] Sack, J.R.; Urrutia, J.. *Handbook of Computational Geometry*. Elsevier Science B.V. 2000.

[TE84] Toussaint, G.T., ElGindy, H.A. *Separationof two monotone polygons in linear time* , Robotica, Vol.2, 215-220, 1984.

[Tou85] Toussaint, G.T. *Computational Geometry* Edited by North-Holland, Amsterdam, 1985 .

[Tou89] Toussaint, G.T. *On Separating two simple polygons by a single traslation.* Discrete and Computational Geometry, Vol.4, 265-278, 1989.

[Tou92] Toussaint, G.T. *What is Computational Geometry?* Proc. IEEE, vol. 80, No. 9, pp. September,1992,1347-1363.

[Wel88] Welzl, E.; *Partition Tree for triangule counting and other range searching problems*; In Proc. 4 ACM Sympossim on Computational Geometry, 23-33,1988.

[Wil82] Willard, D.E.; *Polygon retrieval;* SIAM J. Comput., 11;149-165;1982.