**Introducción**

Cuando se tienen datos de puntos multidimensionales, estos se pueden representar de distintas maneras. La decisión sobre qué tipo de representación utilizar se verá fuertemente afectada por el tipo de operaciones que se ejecutarán sobre estos datos. Para el problema propuesto se debe tener especial cuidado con el uso de memoria, los tiempos de acceso a datos y la capacidad de tener datos dinámicos, los que puedan ser insertados o removidos de la estructura sin perjudicar al resto de los datos.

Los registros de datos espaciales suelen incluir la localización de los datos como uno de sus atributos. Para el caso de datos que se encuentran en un espacio bidimensional, es necesario hacer uso de estructuras de datos que permitan accesos eficientes a los datos para inserción y búsqueda. Un tipo de estructura que permite esto, es el de los Quadtree, que corresponde a un árbol de nodos que subdividen el espacio.

**Estructuras**

**QuadTrees**

El Quadtree corresponde a una estructura de tipo árbol en la que cada nodo posee 4 nodos hijo. Es el análogo bidimensional del Octree y es utilizado para particionar un espacio bidimensional de forma recursiva subdividiéndolo en 4 cuadrantes. Los datos pueden estar almacenados como estructuras los que se asocian a las hojas del árbol. Existen distintas clases de Quadtree, las que pueden poseer subdivisiones cuadradas o rectangulares dependiendo del tipo. En general las estructuras de tipo Quadtree cumplen con las siguientes propiedades:

* Subdividen el espacio en regiones referenciadas por nodos.
* Cada región o nodo posee una capacidad máxima de almacenamiento de puntos, el cual al ser sobrepasado fuerza a una subdivisión del espacio y a la creación de nuevos nodos.
* Existe una relación directa entre la topología del árbol creado y la relación espacial que existe entre los datos.

Los Quadtree se pueden clasificar en función del tipo de datos que representan, tales como puntos, líneas, curvas y áreas. Algunos tipos son el Region Quadtree, el Point Quadtree, Point-region Quadtree y Edge Quadtree.

**Point-Region Quadtree**

Un PR-Quadtree corresponde a un tipo de Quadtree donde cada nodo puede ser un nodo padre, el cual posee exactamente 4 nodos hijo, o es un nodo hoja. Esta estructura representa una colección de puntos en un espacio bidimensional los que se encuentran relacionados a datos. Así, si una región contiene un solo punto o no contiene ningún punto, este espacio se puede representar en el árbol mediante una hoja. Cuando una región contiene más de un punto, es necesario subdividir el espacio en cuatro partes de forma recursiva hasta que estos puntos se encuentren en nodos hoja separados. De esta forma, se tiene que el PR-Quadtree es un árbol cuyos nodos representan una región del espacio y también pueden contener subárboles de subregiones del espacio. A estos cuadrantes o subregiones se les puede llamar NW, NE, SW, SE.

A diferencia de estructuras como los árboles binarios, donde la estructura del árbol no solo depende de los datos insertados, sino que también del orden en el que se insertan, la estructura que toma un PR-Quadtree depende solamente de los datos que contiene, sin importar el orden ya que esta estructura reorganiza las regiones donde se insertan o remuevan nuevos puntos en función de la localización de los puntos.

Una de las principales razones para usar una estructura PR-Quadtree se debe a los tiempos de acceso a los datos. Datos que se encuentren bien distribuidos en el espacio resultarán en árboles de menor profundidad, y los datos se encuentran almacenados en las hojas del árbol por lo que resulta beneficioso tener un árbol de la menor profundidad posible. El Quadtree no es un árbol simétrico, ya que su topología depende directamente de la posición espacial de los datos, debido a esto es posible tener casos donde algunos puntos se encuentren muy cercanos en el espacio, lo que se vería reflejado en ramas muy profundas en el Quadtree, lo que resulta en tiempos de acceso mayores. En cierta forma la estructura PR-Quadtree se puede utilizar como una especie de Tabla Hash, donde se puede ocupar una coordenada bidimensional como llave al dato.

A picture containing bird

Description automatically generated

A close up of a device

Description automatically generated

Para implementar una estructura PR-Quadtree es necesario primero representar sus partes. Una forma simple de hacer esto es considerar distintos tipos de nodo, que se diferencien en la cantidad de puntos que contienen en el cuadrante que representan. De esta forma se tienen 3 tipos de nodo, los nodos blancos, los nodos negros y los nodos grises. Los nodos blancos corresponden a aquellos nodos hoja que no contienen ningún punto en su interior, los nodos negros son aquellos nodos hoja que contienen un solo punto y los nodos grises son nodos que contienen más de un punto y por lo tanto son nodos intermedios (no hoja) que apuntan a 4 nodos hijo, los cuales a su vez pueden también ser blancos, negros o grises. La estructura PR-Quadtree al iniciar vacía consiste solo de un nodo blanco en un principio. Así, es posible representar el PR-Quadtree de la siguiente manera mediante estructuras:

Nodo:

Data\* dato

Nodo\* cuadrante\_1

Nodo\* cuadrante\_2

Nodo\* cuadrante\_3

Nodo\* cuadrante\_4

Data:

dato

pos x

pos y

PR-Quadtree:

Nodo\* root

La inserción de puntos en la estructura PR-Quadtree tiene que cumplir con

Colisiones

La eliminación de datos del PR-Quadtree

Compactacion

La búsqueda en estructuras PR-Quadtree

**Conclusión**

* Las PR-Quadtree se pueden utilizar como hash-tables de llave bidimensional.
* Se puede mejorar el rendimiento almacenando mas de un punto por cuadrante
* Se podría hacer algo análogo a las funciones Hash, donde antes de aplicar un insert se aplicase una función a los puntos de forma que estos tengan una transformación en el espacio. Si se utilizara una transformación que distribuyese mejor los puntos en el espacio (como para casos en que existan zonas muy densas) se podría obtener un árbol de nodos mejor distribuidos, es decir ramas menos profundas, lo que impactaría directamente en la velocidad de acceso a los datos.

<https://opendsa-server.cs.vt.edu/ODSA/Books/CS3/html/PRquadtree.html>

<http://courses.cs.vt.edu/~cs3114/Summer15/Notes/T04_PRQuadTrees.pdf>

<https://es.wikipedia.org/wiki/Quadtree>

<http://www.cs.umd.edu/~hjs/pubs/AngSSD89.pdf>