RTREE

Daniel Palomino,
Jose Chavez,
Vittorino Mandujano,
Wilderd Mamani

Agosto 15, 2018

Resumen

El presente reporte abarca la implementación y el funcionamiento de un **R-Tree**. El **R-Tree** es una estructura de datos multidimensional usado para metodos de acceso espacial. Algunos ejemplos de estas estructuras son: sistemas de coordenadas geográficos, rectángulos o polígonos. Esta estructura es de mucha importancia por las aplicaciones teóricas y prácticas en el mundo real, como por ejemplo "Encontrar todos los museos en un radio de dos kilómetros alrededor de la posición actual".La idea principal de esta estructuda de datos y la descripción será descrita así como la implementación en un demo web con una interface gráfica. El demo implementado nos permite dibujar puntos y polígonos con el mouse en un entorno canvas descrito en este reporte. Adicionalmente se presentan los resultados obtenidos, las limitaciones así como las conclusiones y trabajos futuros.

Documentación adicional:

1. GitHub: `https://github.com/AED-RTREE/rtree`.
2. Video Youtube: `https://www.youtube.com/watch?v=a6DZWz91xNo`

Descripción Conceptual del R-Tree

El **R-Tree** fue creado por Antonin Guttman el año 1984. Puede considerarse como una generalización del B-Tree, una estructura de datos unidimensional. El **R-Tree** es idóneo para representar objetos en un espacio multidimensional, ergo para figuras en el plano.

El **R-Tree** es un árbol balanceado donde todos sus objetos están en las hojas, encerradas por una región. Para el caso bidimensional, los objetos son puntos y polígonos delimitado

por un cuadrilátero. Los nodos de la parte superior están relacionados con las regiones que contienen a dichos objetos. Cada nodo posee una cantidad bien diefinida de regiones y en cada región hay un nodo. La forma del árbol resultante podemos contemplarla en la figura 1.

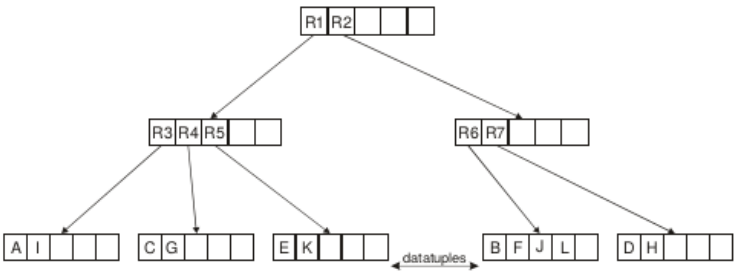

Figure 1: Ejemplo de un RTree.

Cada nodo tiene una cantidad mínima m y máxima M de regiones. Una cantidad fuera de dicho rango ocasiona, para un exceso de regiones, un split o división del nodo. Por otra parte, una cantidad menor al rango ocasiona una unión de estos. La cantidad mínima de regiones suele ser cerca de la mitad de la cantidad máxima de regiones, la única excepción a dicha regla ocurre con el nodo raíz que puede tener un minimo de 2 regiones.

Tomando por ejemplo el caso basado en los datos de la figura 2. Si lo representamos con un R-Tree con un $m = 2$ y un $M = 5$ tendríamos un árbol como en la figura 3.

name	semester	credits
A	8	100
B	4	10
C	6	35
D	1	10
E	6	40
F	5	45
G	7	85
H	3	20
I	10	70
J	2	30
K	8	50
L	4	50

Figure 2: Ejemplo de base de datos alumno y creditos

Podemos contemplar que la raíz esta formada por las regiones $R1$ y $R2$. Estas a su vez apuntan a los nodos del siguiente nivel. El nodo que apunta $R1$ contiene las regiones $R3$, $R4$ y $R5$ y el nodo que apunta $R2$ contiene las regiones $R6$ y $R7$. Este árbol podemos representarlo gráficamente en el plano tal como se muestra en la figura 4.

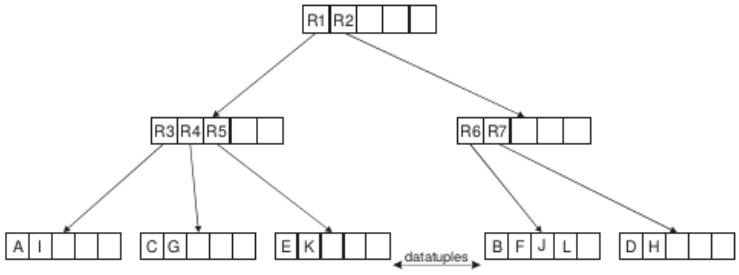

Figure 3: RTree correspondiente a los datos de la Fig. 2

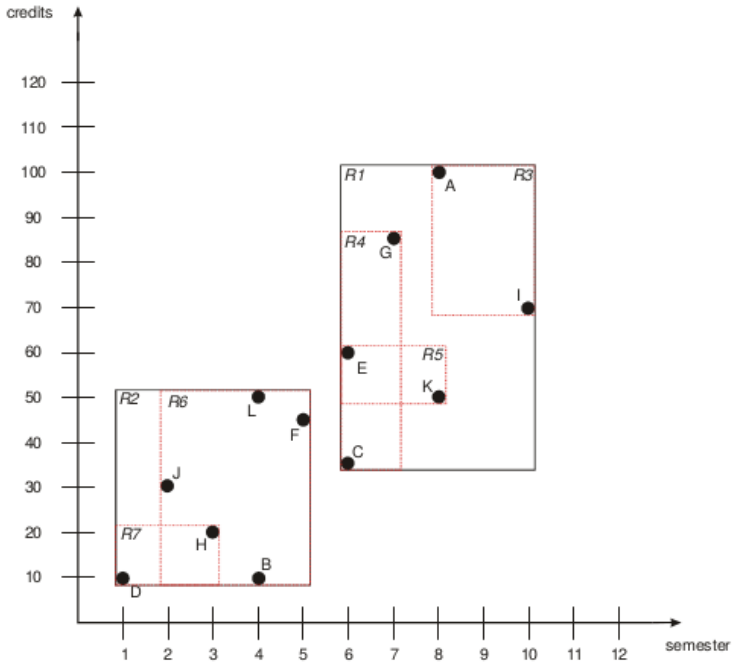

Figure 4: La grafica muestra el diagrama de los rectangulos de la base de datos ejemplo

Como toda estructura de datos, en el R-Tree podemos realizar las operaciones elementales: Búsqueda, Inserción y Eliminación.

Busqueda: la busqueda en el R-Tree funciona de manera similar al B-Tree, tal y como este ultimo el arbol va descender desde la raiz, en los arboles R se almacenan rectangulos quienes se pueden sobreponer, por que todos los rectangulos tienen que ser visitados. Sea T la raiz de un r-tree. ahora vamos a buscar todos los indices guardados de quienes los rectangulos un rectángulo de búsqueda especificado S,

- si T no es una hoja, entonces aplicamos la busqueda en cada hijo de quienes la raiz esta apunta por el puntero hijo y este se superpone con S.
- Si T es una hoja, retorna todas la entradas que se superponen con S como el conjunto resultante.

un ejemplo de acuerdo a nuestro ejemplo base seria graficamente.

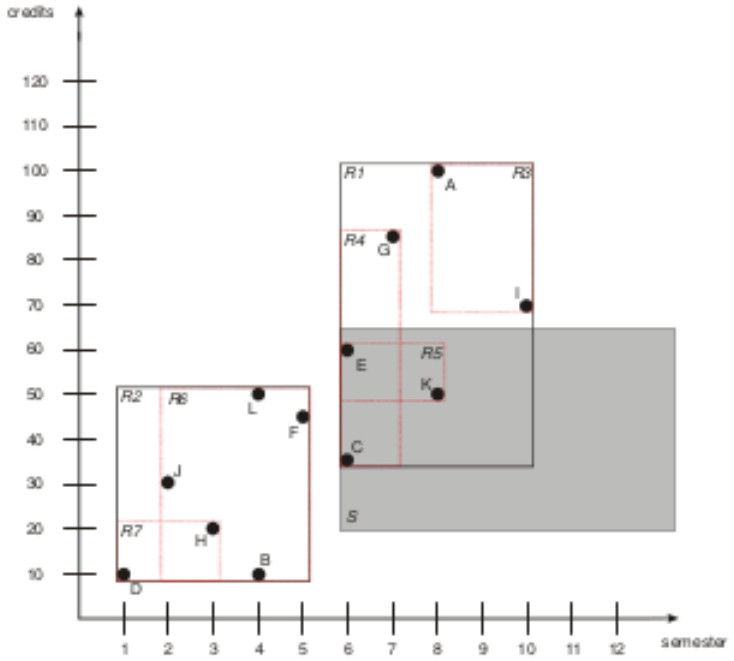

Figure 5: Ejemplo de busqueda de estudiantes.

En el ejemplo anterio queremos encontrar todos los estudiantes que estan en el secto semestre o mas y que tienen entre 20 y 65 creditos, ademas podemos ver que R1 se superponecon la consulta del rectangulo S,sin R2, asi que vamos a buscar en R1, en el siguiente paso R4 y R5 esta superpuestos con S en estos rectangulo de busqueda encontramos que C esta dentro de R4 y E,K estan dentro de R5, asi el resultado es C, E, K

Inserción: Se tiene un objeto (polígono o punto) y se delimita por un MBR. Luego se realiza el proceso de búsqueda para añadirlo en las regiones y nodos correspondientes. Después se tiene que verificar si el árbol todavía cumple las condiciones impuestas por m y M. Se realiza dicha prueba por el camino que fue descrito por la búsqueda, pero en orden inverso.

Eliminación: Para eliminar un objeto se realiza la búsqueda del objeto. Una vez encontrado este es eliminado y se revisa los nodos por los que se recorrió en la búsqueda para cumplir las condiciones de m y M.

Implementacion del Demo web

Para la implementación del presente Demo Web, se usó una arquitectura de 3 capas:

1. La Aplicación Web, desarrollado en un entorno canvas de html5 y Javascript para la funcionalidad y manipulación de las etiqueta.

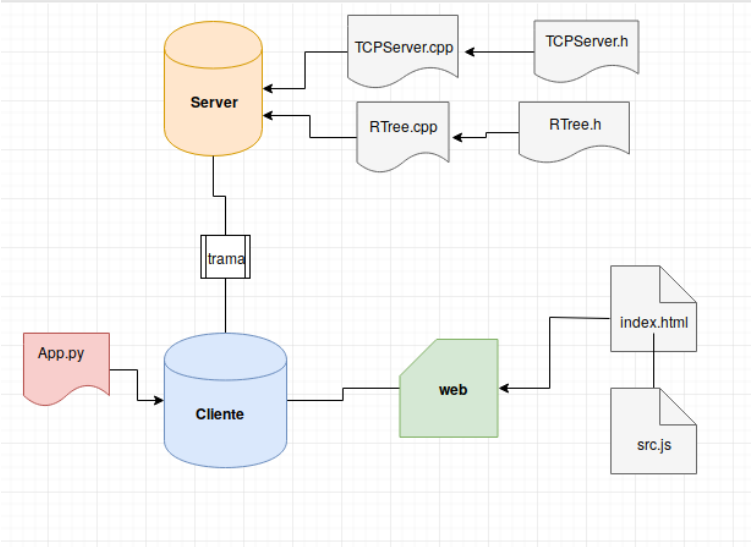

Figure 6: Diagrama general del demo web

2. La conectivad basada en sockets, siendo el servidor construído sobre c++ con métodos de bajo nivel implementados sobre c con librerías nativas de sistemas tipo unix como socket.h, y el ciente construído sobre python.

3. El Aplicación RTree, que almacena los objetos insertados por el usuario, para posteriormente poder realizar consultas sobre él.

En la *Figura*[6] se puede apreciar el flujo de información a través de estos 3 níveles o capas.

Conectividad

La capa de conectividad está compuesta por 2 partes:

1. El Socket Servidor, implementado en c++ usando la librerías *socket.h* nativa de sistema tipo UniX.

2. El Socket Cliente, implementado en python 2.7 usando el módulo *socket* de su libería estándar.

Para un mejor manejo de la información entre ambos procesos se diseñó un protocolo simple para el envío de los objetos insertado y las diferentes consultas posibles (Range, Nearest) y una respuesta que permita saber si la información fue envíada y recibida de manera correcta, y prevenir de esta manera errores por pérdida de datos.

En la *Figura*[7] se muestra un ejemplo de algunos casos usando este protocolo.

```
QUERIES:
* INSERT|1|100|230|END
* RANGE|2|120|360|520|450|END
* NEAREST|K|1|120|320|END
* DELETE|0|END

RESPONSES:
* DELETE|0|END
* SUCCESS|0|END
* FAIL|0|END
* EXIT|0|END
```

Figure 7: Trama de consultas y respuestas del TCPServer

El funcionamiento es bastante simple, se construye un *string* usando el *comando* envíado más los objetos a enviar, todo separado por palotes (|).

Siendo la estructura general de envío desde el cliente de la siguiente manera:

$$\text{COMMAND}|N|X_1|Y_1|X_2|Y_2 \ldots |\text{END}$$

Donde:

1. **N** es el numero de puntos del objeto a enviar.
2. Finalmente, las coordenadas de los puntos del objeto en orden $(x_1, y_1) \ldots (x_c, y_C)$.

Y la respuesta del servidor:

$$\text{COMMAND}|N|M_1||X_1|Y_1|X_2|Y_2 \ldots |\text{END}$$

Donde:

1. **N** es el número de niveles del árbol.
2. **M** es el número de MBRs de cada nivel.
3. seguido del número de **C** puntos a enviar por MBR.
4. Finalmente, coordenadas de los puntos del MBR en orden $(x_1, y_1) \ldots (x_c, y_C)$.

Aplicación Web

La implementan web fue desarrollada con la tecnología HTML5 y Javascript para la manipulación de las etiquetas html y el entorno gráfico, en el cual se pueden insertar polígonos y los puntos, para esto se utilizó canvas-html y para crear la aplicación web se utilizó Flask, un framework de Python. Además se aprovechó el módulo "socket" de la librería estándar para la conexión con el servidor. De este modo logramos la conexión con nuestro código fuente desarrollado en c++, los detalles de instalación y ejecución están detallados en un repositorio GitHub.

Aplicativo cuenta con la función INSERT, que nos permite ingresar puntos y poligons, dentro de un recuadro de 854x480 pixeles, permitiendo visualizar en tiempo real los MBRs generados con su respectiva etiqueta y color. Las etiquetas siguen un orden específico de acuerdo al nivel, empezando desde la vocal 'A' y adjuntando el número respectivo de MBR en el nivel, del mismo modo los colores ayudan a diferenciar los niveles.

La función RANGE permite encontrar los elementos dentro de un rectángulo, y resaltando los elementos encontrados en color verde. Por ultimo se tiene la función NEAREST que nos permite encontrar los elementos mas cercanos a un punto, previamente ingresando un valor (k), en otras palabras encuentra los k-elementos más cercanos a un punto. Los elementos son resaltados del mismo modo, con un color verde y una respectiva linea que permite visualizar de una mejor manera a que elementos se refiere.

Por último se tiene la función DELETE ALL que elimina todos los poligonos y puntos y por consecuencia todos los MBRs generados.

Cada una de estas funciones fueron diseñadas en Javascript, estas funciones envían los datos a un programa en Python que se encarga de empaquetar los datos y enviarlos al Servidor con la ayuda de la librería 'socket' de Python. Este programa también recibe los datos del servidor y se encarga de modificar los valores en Javascript para luego dibujarlos dentro del canvas y asi poder visualizar las diferentes figuras.

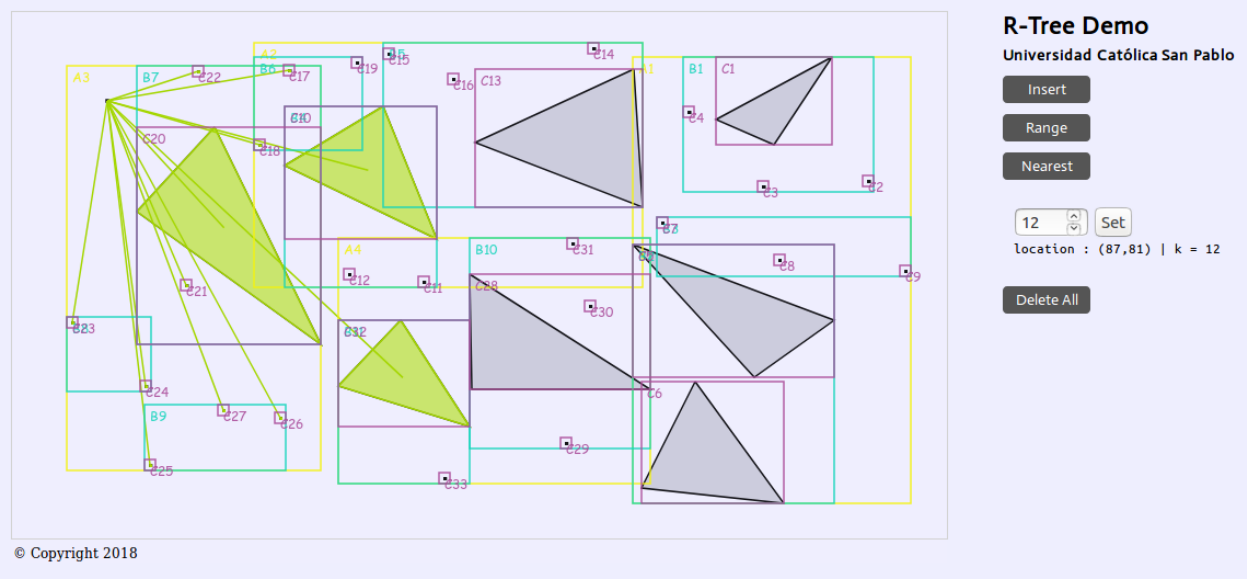

Figure 8: Ejecutando la funcion K-Neasrest

Implementacion R-Tree:

Estructura del R-Tree: El R-Tree fue implementado con la clase R-Tree y varias estructuras, quienes son: Rect, Node, Branch, ListNode y PartitionVars.

- Nodo: La clase nodo, la clase RTree tiene por defecto uno y es la raíz
- Region: corresponde a las regiones que poseen los nodos. Las regiones tienen un puntero a Nodo.
- Mbr: Corresponde al MBR de las regiones, todas las Regiones tienen uno.
- ListNode: Lista que contienen nodos y sirven para grabar el recorrido de la búsqueda.
- PartitionVars: Estructura que facilita el split de los nodos .

Descripción del Código : Consta de 5 funciones principales correspondientes a insertar,eliminar todos, búsqueda y busqueda de los objetos cercanos. Y de funciones complementarias que son 34.

Resultados

En este informe detallamos la implementación de un demo web, con visualización grafica de los poligonos con regiones sombreadas y puntos, a los cuales podemos hacer consultas usando el mouse como las regiones a un rectángulo y los puntos mas cercanos a un punto k elegido. El demo muestra la partición a medida que se va insertando mas poligonos, asi como las regiones MBR que contienen a estas de tal manera que la respuesta del algoritmo R-Tree implementado en c++ es visualizada en la pantalla canvas del demo. Además podemos ver de manera interactiva los puntos mas cercanos, atraves de una linea a todos los puntos mas cercanos del punto elejido con el mouse y las figuras coloreadas a una región (rectángulo) seleccionado por el mouse. Finalmente podemos ver los puntos en coordenadas que usamos con el mouse en cada elección de los botones, de tal manera que podemos ver los puntos elegidos en cada click. Asi en el demo web se puede apresiar de manera visual la estructura de datos espacial R-Tree.

Limitaciones

- El presente proyecto de aplicación web de la estructura de datos multidimencional R-Tree ha sido desarrollada para un caso particular de poligonos y puntos, por fines de estudio y aplicacion práctica de la teoria del algoritmo.
- El Algoritmo modulo cuenta con un modulo split utilizado para la particion el cual cuenta con uno de las muchas formas de particion con fines practicos se utilizo la particion cuadratica.
- La ejecución e instalacion del presente proyecto necesita un sistema ubuntu del parte del servidor por la dependencia de las librerias.

Trabajos Futuros

- Implementar la aplicación solo en Javascript de tal modo que se pueda ejecutar las funciones de forma continua y ver los MBRs en tiempo real.
- Visualizar el árbol completo de forma dinámica.
- Permitir la configuración de los parámetros del R-Tree desde la aplicación web.
- Implementar la aplicación para la ejecucion del demo web y eliminar la dependencia del sistema operativo.

Conclusiones

Este proyecto muestra la implementación de un R-Tree en un vizualizador web canvas de poligonos y puntos, para fines netamente aplicativos de la estructura de datos R-Tree,a pesar que nuestro demo muestra los puntos pequeños, añadimos una vizualización mas interactiva de acuerdo a movimiento del mouse, por lo cual podemos mostrar como funciona el k nearest. Por otro lado la implementación de la parte del servidor se realizó con libreria unix , por lo que recomendamos usar linux ubuntu para test del demo, debido a la dependencia que encontramos en la librerias antes mencionada en el lado del servidor de extension cpp.

Referencias

1. *Spatial Data Structures class*; Laura Toma; `http://www.bowdoin.edu/~ltoma/teaching/cs340/spring08/Papers/Rtree-chap1.pdf`
2. *R-Trees: Theory and Applications*;Y. Manolopoulos; A. Nanopoulos; Y. Theodoridis (2006)
3. *R-Tree concept* ; wikipedia the free enciclopedia; `https://en.wikipedia.org/wiki/R-tree`
4. *R-trees: a dynamic index structure for spatial searching*; Dan Hanmore Antonin Guttman University of California, Berkeley;
5. *R-Tree concept* ; wikipedia the free enciclopedia; `https://en.wikipedia.org/wiki/R-tree`
6. *The Priority R-tree*; `http://www.win.tue.nl/~mdberg/Papers/prtree.pdf`

Distribución de trabajo

- **Daniel Palomino:** Socket-Server, RTree.
- **José Chávez:** Aplicación Web, Socket-Cliente.
- **Vittorino Mandujano:** RTree.
- **Wilderd Mamani:** Aplicación Web.