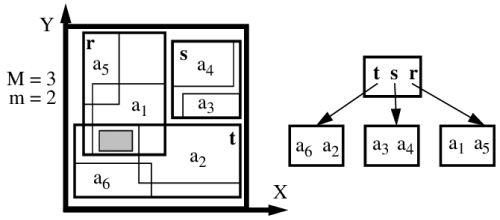
Tarea 1: R-Trees

Profesores: Benjamín Bustos
Gonzalo Navarro
Auxiliares: Asunción Gómez
Diego Salas

1 Introducción

Un R-tree es un árbol, parecido al B-tree, que permite manipular rectángulos en memoria secundaria, además de soportar tanto la inserción como la eliminación de datos. Una consulta de búsqueda dentro de este es de la forma: dado un rectángulo C, se deben retornar todos los rectángulos del R-tree que se intersectan con C. En la figura 1 se muestra un ejemplo de la estructura.

Figure 1: Ejemplo de un R- tree. Fuente: researchgate.



En esta tarea se estudiarán formas de construir un R-tree a partir de un conjunto amplio de datos, sin necesariamente utilizar métodos de inserción para hacerlo, sino, utilizando ciertos parámetros que nos permitan acercar a una distribución óptima del árbol.

2 El R-tree

Un R-tree debe cumplir las siguientes propiedades:

- Cada nodo interno representa conceptualmente un rectángulo: el *Minimum Bounding Rectangle* (MBR) de los rectángulos nodos que cuelgan de él. De esta forma, al hablar de un nodo también hablamos de un rectángulo.
- Los rectángulos correspondientes a datos se encuentran almacenados en las hojas del árbol.
- Cada nodo de un R-tree almacena físicamente a los más M hijos.
- Todos los nodos hojas están a la misma profundidad en el árbol.

La estructura del R-tree contiene en sus nodos k claves y k hijos. Las claves son una 4-tupla que define al rectángulo: x_1, y_1, x_2, y_2 y que permiten conocer el área del rectángulo. Los hijos

apuntan a los rectángulos contenidos en aquel rectángulo definido por la clave (como se vio en la figura 1). Para más información sobre el R-tree, puede revisar el apunte en la página 47.

3 Objetivos

Para esta tarea, se deberá implementar:

1. Una función por cada método de construcción, que permita, a partir de un conjunto de rectángulos, construir el árbol.
2. El método de búsqueda dentro del árbol.
3. Evaluar el costo del método de búsqueda con los distintos métodos de construcción.

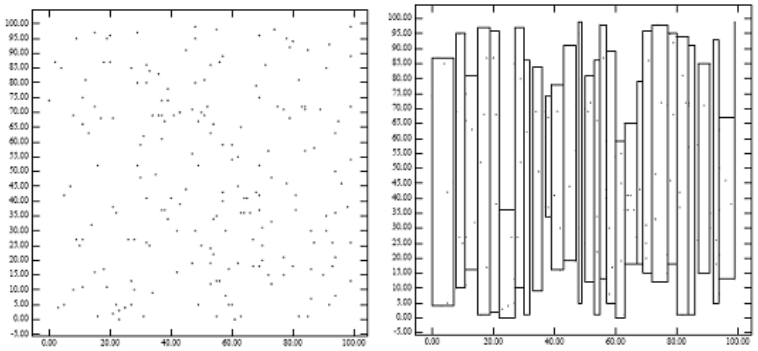
De esta forma, la idea es hacer una serie de búsquedas en el R-tree y ver cuál método de construcción funciona mejor.

3.1 Construcción

3.1.1 Primer método de construcción: Nearest-X

Dado un conjunto R de n rectángulos, y M la máxima cantidad de hijos de cada nodo del árbol, el método ordena los rectángulos de R según la coordenada X del centro del rectángulo, para luego juntarlos en grupos de tamaño M, usando rectángulos consecutivos en la secuencia ordenada, para con ellos formar $\frac{n}{M}$ nodos con M hijos cada uno (el último nodo puede quedar con una menor cantidad de hijos). De estos $\frac{n}{M}$ nodos, identificados por rectángulos (el MBR que contiene a todos sus hijos), tomamos nuevamente el centro y repetimos la anterior forma de agrupar los puntos con este nuevo conjunto de nodos. Esto se repite hasta que se puedan agrupar todos los nodos de una iteración en un solo nodo raíz.

Figure 2: Conjunto de puntos agrupados según su coordenada X. Fuente: Wikipedia.

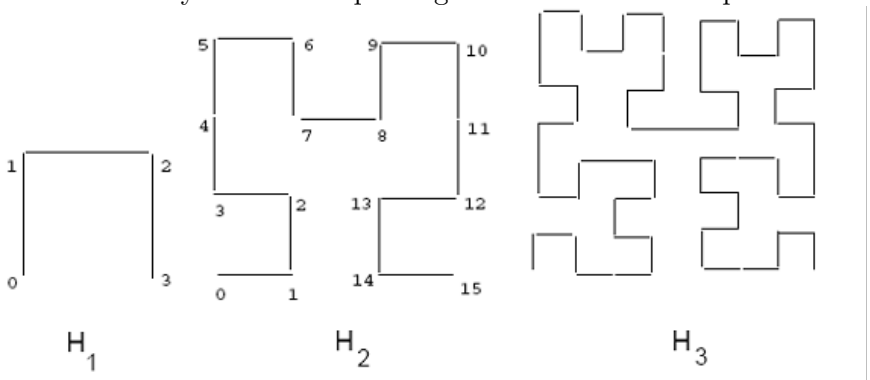


En el ejemplo, se pueden considerar los puntos como rectángulos de área 0, es fácil notar que se forman segmentos largos verticalmente, dado que los puntos se agrupan según su coordenada X.

3.1.2 Segundo método de construcción: Hilbert R-tree

Es un método similar al primero, solo que ordena el centro de los rectángulos en función de su valor dentro de la Curva de Hilbert. La lógica recursiva de armar el árbol con nodos de M hijos hasta llegar al nodo raíz se mantiene, siendo lo único que cambia la forma en la que se ordenan los nodos, en este caso, los puntos de referencia se ordenan según su valor en la curva de Hilbert, la cual es una curva fractal que cubre el plano, y puede asignar valores a cada punto en función de cuanta distancia hay desde el inicio a este.

Figure 3: Curvas de Hilbert y los valores que asigna a cada valor en el plano.Fuente: Wikipedia.

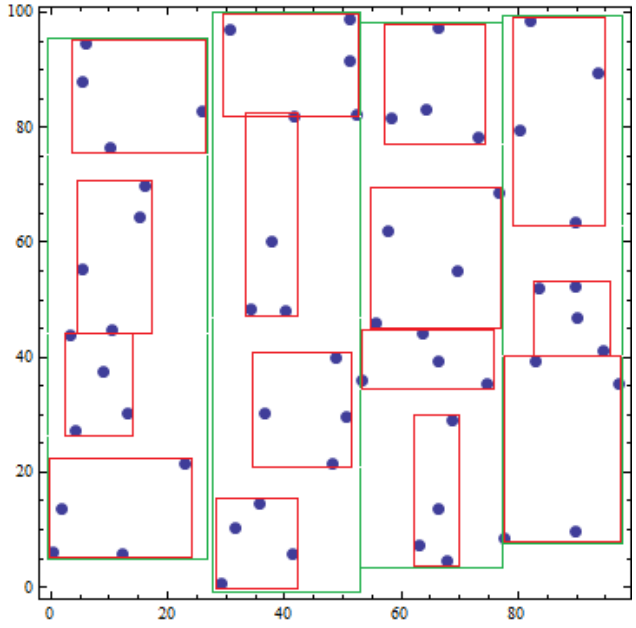


Siendo el plano que cubre cada curva de Hilbert una potencia de 2, asigne los valores según la menor curva que cubra todo el plano al que pertenecen los puntos de referencia contenidos en el árbol. Tanto para este árbol como para el siguiente, al final del enunciado se referencia un paper que explica los métodos de manera detallada.

3.1.3 Tercer método de construcción: Sort-Tile-Recursive

El método comienza ordenando inicialmente los n puntos de referencia según su coordenada X, dividiendo el resultado en $S = \sqrt{\frac{n}{M}}$ grupos con $M \cdot S$ nodos cada uno, donde $\frac{n}{M}$ viene a ser la cantidad necesaria de hojas en el árbol de forma que se almacenen todos los nodos. Cada uno de los S grupos se ordena nuevamente según su coordenada Y, y luego se divide en S grupos de tamaño M. Con esto, se terminan formando $S \cdot S = \frac{n}{M}$ nuevos nodos, que serán agrupados de forma recursiva siguiendo el mismo método, reduciendo constantemente la cantidad de nodos hasta poder unirlos todo en un único nodo raíz.

Figure 4: Ejemplo de STR en un conjunto de puntos.



En el ejemplo ($n = 60$ y $M = 4$), se separa primero la coordenada X de los puntos de referencia en $S = \sqrt{\frac{60}{4}} \approx 4$ segmentos te tamaño $S \cdot M = 16$ (a excepción del último), esta división se puede visualizar con los rectángulos de color verde (ojo que estos rectángulos no son MBR, y no pertenecen al R-tree, solo los rectángulos rojos). Luego, por cada una de estas divisiones, se hacen $S = 4$ grupos de $n = 4$ puntos.

3.2 Búsqueda

Dado un rectángulo C, la operación de búsqueda recorre el árbol para encontrar todos los rectángulos en los nodos hojas que lo intersectan. Si en algún nodo interno, un MBR no intersecta a C, se pueden descartar los puntos pertenecientes a ese subárbol como parte de la solución. Finalmente, cuando se llega a una hoja, solo queda verificar los rectángulos que se intersectan con el rectángulo

de la consulta. Al implementar el método, este deberá devolver una lista con todos los rectángulos pertenecientes a la solución, además de calcular la cantidad de accesos a bloques de disco realizados durante su ejecución.

4 Experimentación

Escoja el parámetro M de acuerdo a las características de su máquina, de manera que un nodo quepa siempre en una página de disco. Documente las características de su máquina, el sistema operativo, lenguaje y compilador utilizados, RAM y características del disco duro.

En los experimentos se pide comparar el rendimiento de cada construcción del R-tree dado el mismo set de datos, el R-tree debe estar almacenado en memoria secundaria y el set de consultas debe ser el mismo por cada árbol. No pueden haber partes del árbol en memoria RAM, todo el árbol debe estar en disco. Antes de cada consulta, ejecute con `sudo` el siguiente comando (linux):

```
echo 3 > /proc/sys/vm/drop_caches
```

para que el sistema operativo no cachee el archivo con el R-tree.

En cada experimento, se pide comparar:

- El tiempo de búsqueda promedio.
- La cantidad de accesos a bloques de disco de la búsqueda.

Tanto para el set de rectángulos R, como para el set de consultas Q, se debe utilizar valores aleatorios enteros uniformemente distribuidos en el rango de $[0, 500000]$, el tamaño de cada lado de los rectángulos de R debe estar uniformemente distribuido entre $[0, 100]$, mientras que el tamaño de cada lado de los rectángulos de Q debe estar uniformemente distribuido entre $[0, 100000]$. Evalúe el resultado para cada $n \in \{2^{10}, 2^{11} ... 2^{25}\}$. Ocupe el mismo conjunto R y Q para evaluar los tres métodos de construcción del R-tree. Realice 100 consultas por cada valor de n, y reporte el intervalo de confianza de sus resultados.

A partir de lo anterior, construya un gráfico que compare los resultados entre sí. El tiempo de búsqueda en un R-tree suele ser de la forma $O(c * n^{\alpha})$ para algún $0 < \alpha < 1$. Ajuste esta curva con los resultados obtenidos para estimar el valor de c y α, y discuta qué tan buena es la aproximación.

5 Entrega

- La tarea puede realizarse en grupos de a lo más 3 personas.
- Para la implementación puede utilizar C, C++ o Java.

- Para el informe se recomienda utilizar Latex.
- Escriba un informe claro y conciso, la entrega debe contar con el código de su implementación y todas las indicaciones necesarias para su ejecución explicadas de forma clara.
- La nota del informe se calculará de la forma:
  - Introducción: 1 ptos.
  - Desarrollo: 1 ptos.
  - Resultados: 2.6 ptos.
  - Análisis: 1.4 ptos.
  - Conclusión: 1 ptos.

6 Referencias

- Paper original R-trees.
- Paper con una explicación detallada del Hilbert R-tree y STR.