Taller 4: Arboles de partición

Juan Esteban Muñoz Diaz, Santiago Rueda Pineda, Victor Julio Peñaranda Florez

Estructuras de datos, Pontificia Universidad Javeriana,
Bogotá, Colombia

Abstract: En el ámbito de la computación gráfica, los árboles de partición son una estructura de datos muy utilizada para representar información geométrica. Un tipo particular de árbol de partición, conocido como Quadtree, es utilizado para dividir un espacio bidimensional en subregiones cuadradas más pequeñas. La creación de un Quadtree típicamente involucra la partición recursiva del espacio en subregiones cuadradas hasta que se cumpla cierto criterio de terminación. En este contexto, el recorrido en preorden se refiere a un método de recorrer el árbol en el que se visitan primero los nodos raíz, seguidos por sus hijos izquierdos y derecho. En este artículo se describe un método para crear un Quadtree a partir de un recorrido en preorden del árbol. El algoritmo comienza por construir la raíz del árbol y luego recursivamente construye cada uno de los subárboles a partir de los nodos visitados en el recorrido en preorden.

Index Terms: QuadTree, arboles de partición, comparación de tamaño.

1. TAD NodoQuadTree

1.1. Datos mínimos

- color, caracter, almacena el color del nodo.
- hijos, apuntador de arreglo de NodoQuadTree, almacena las direcciones de memoria de los hijos del Nodo.

1.2. Funciones

- NodoQuadTree(), crea un objeto de tipo NodoQuadTree.
- setColor(color), setter del atributo color.
- getColor(), retorna un carácter con el color del Nodo.
- setHijo(pos, nodo), setter para los hijos, recibe cual hijo y un apuntador al nodo.
- getHijo(pos), retorna el apuntador del hijo requerido.
- crearArbol(datos), crea el árbol recursivamente con el recorrido en preorden en una cola como parámetro.
- preOrden(), imprime el recorrido en preOrden del árbol generado.
- rellenar(minX, maxX, minY, maxY, color, matriz), rellena de un color el sector determinado y lo retorna.
- matrizNodo(xInf, xSup, yInf, ySup, matriz), genera la matriz de valores del árbol y lo retorna.

2. TAD QuadTree

2.1. Datos mínimos

- raiz, NodoQuadTree, apuntador al nodo Raíz del árbol.

2.2. Funciones

- crearArbol(nodo), recibe el apuntador a un NodoQuadTree para generar el QuadTree.

- preOrden, llama a la función preOrden del NodoQuadTree.
- matrizArbol(arch, ancho, alto), genera la matriz del QuadTree.

3. TAD descomprimir

3.1. Datos mínimos

- arbol, QuadTree, almacena el QuadTree que se usará.
- entrada, cola de caracteres, almacena el preOrden a usar.

3.2. Funciones

- procesamiento(argc, argv), realiza el procesamiento del archivo recibido para generar el archivo .pbm con el nombre ingresado.

4. Contenido de los archivos preOrden

- image_00.qt: La imagen tiene en el cuadrante 1 y 3 blanco y en el 2 y 4 negro, similar a un tablero de ajedrez.
- image_01.qt: La imagen tiene fondo negro, con 14 círculos en un tipo de espiral.
- image_02.qt: La imagen tiene un Homero con una cerveza y la frase ”To alcohol!”.
- image_03.qt: La imagen tiene a una persona con la frase ”Thou Shalt not Pass”.
- image_04.qt: La imagen tiene un muñeco con un cuchillo y una pistola botando humo.
- image_05.qt: La imagen tiene un logo similar a una pantera.
- image_06.qt: La imagen tiene un muñeco sacando la lengua similar a un pingüino.
- image_07.qt: La imagen tiene algo similar a una escalera donde el cuadrante 1 esta completamente blanco y el 4 completamente negro.

5. Tamaños de las imagenes

- image_00.qt:
  - En .qt → 1KB
  - En .pbm → 1KB
- image_01.qt:
  - En .qt → 3KB
  - En .pbm → 129KB
- image_02.qt:
  - En .qt → 43 KB
  - En .pbm → 514 KB
- image_03.qt:
  - En .qt → 25 KB
  - En .pbm → 514 KB
- image_04.qt:
  - En .qt → 514 KB
  - En .pbm → 32 KB
- image_05.qt:
  - En .qt → 40 KB
  - En .pbm → 2.051 KB
- image_06.qt:
  - En .qt → 17 KB
  - En .pbm → 514 KB
- image_07.qt:
  - En .qt → 1 KB
  - En .pbm → 1 KB

Con base en los datos presentados, se puede deducir que en la gran mayoria de los casos, es mas óptimo tener almacenada una imagen en la forma de pre-orden de un QuadTree que en la versión de Mapa de bits. Únicamente en imágenes muy complejas, donde no se pueden tener cuadrados de un único color la versión .qt puede llegar a pesar igual que el .pbm.