Organización y manejo de archivos

www.inf.ucv.cl/ wpalma/oma

Dr. Wenceslao Palma
wenceslao.palma@ucv.cl

Índices Multidimensionales

- Los índices vistos hasta el momento son unidimensionales → clave de búsqueda se basa en un solo atributo.
- Supongamos las sgtes consultas:
  - se tienen puntos (x, y) almacenados en un archivo y se requiere recuperar todos los puntos que cumplan con $(100 \leq x \leq 200)$ y $(100 \leq y \leq 300)$ (range query).
  - cuáles son los clientes con $25 \leq edad \leq 35$ y $sueldo \leq 1000K$? (range query).
  - si alguien se encuentra en el punto (x, y), cuáles son los restaurantes más cercanos? (nearest-neighbor query)
- Es posible responderlas con alguno de los índices vistos anteriormente?
- Cuáles son las desventajas?

Índices Multidimensionales

- De acuerdo a las estructuras de datos utilizadas para implementar índices multidimensionales, éstos se pueden dividir en dos categorías.
  - basados en hashing: grid.
  - basados en árbol: *kd*-tree, *quad* tree y árbol-R.

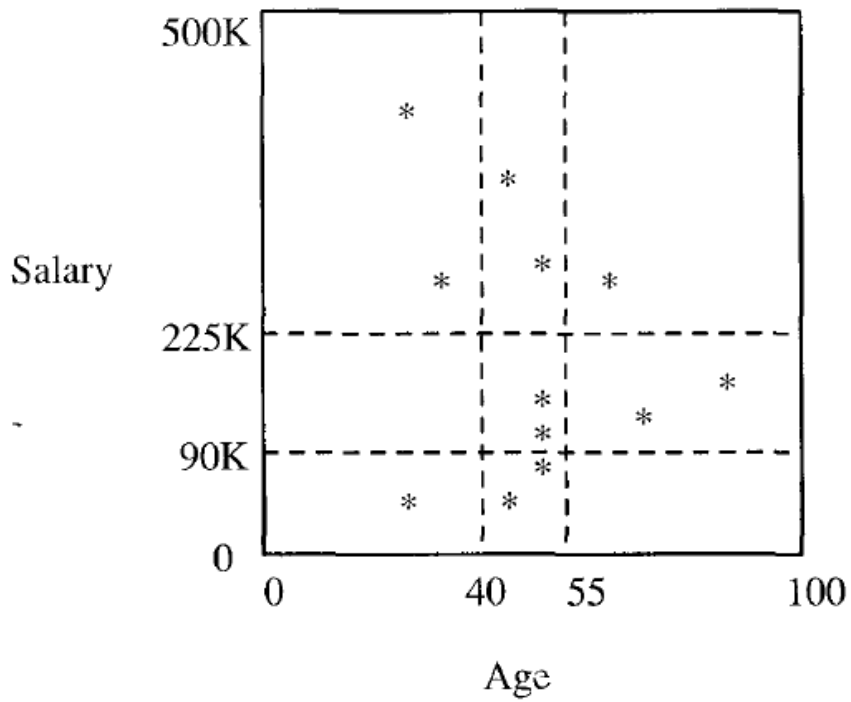
Grid

- Es una estructura de datos simple que generalmente posee mejor rendimiento que índices de una dimensión utilizados para responder consultas sobre múltiples dimensiones.
- En un índice tipo grid, los datos son vistos como puntos los cuales son agrupados en particiones (stripes).
- Las particiones son definidas mediante líneas horizontales/verticales que definen el valor mínimo de dicha partición.
- El número de líneas puede variar y el espacio entre líneas de la misma dimesnión puede ser diferente.

Ejemplo

Los siguientes puntos representan a un conjunto de clientes mediante los atributos $(edad, salario)$:
$(25, 60), (50, 120), (25, 400),$
$(45, 60), (70, 110), (45, 350),$
$(50, 75), (85, 140), (50, 275),$
$(50, 100), (30, 260), (60, 260)$

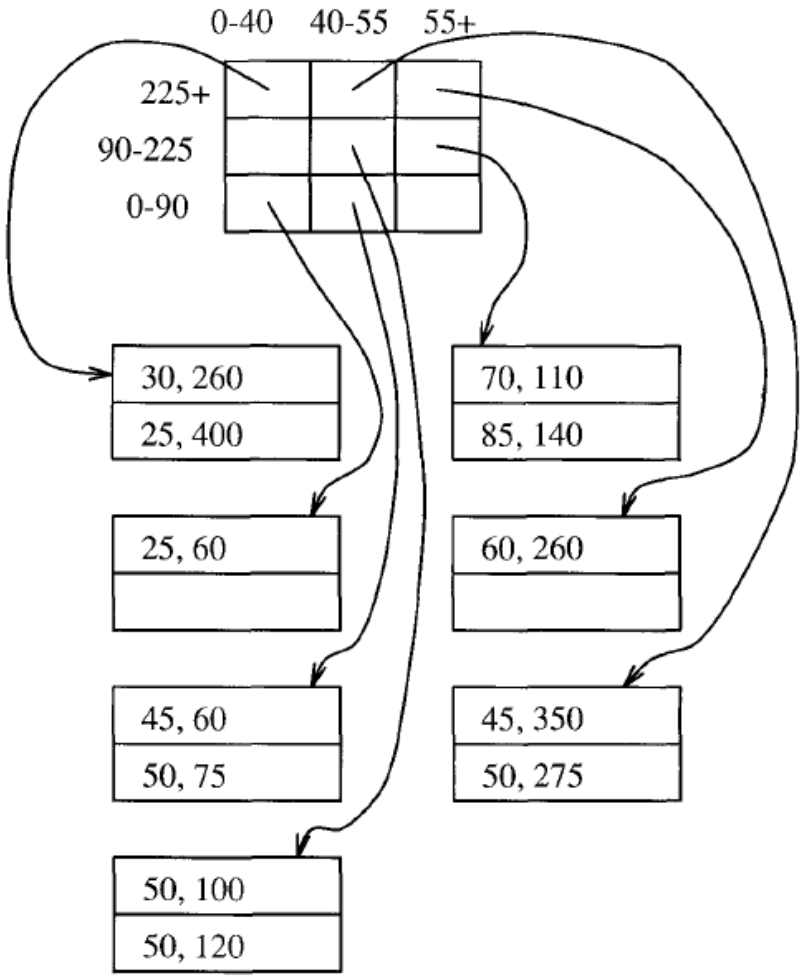
La figura muestra los puntos en un espacio de 2 dimensiones. Las líneas fueron escogidas en forma arbitraria. Rectángulo central de la primera fila representa clientes $40 \leq edad < 55$ y $0 \leq salario < 90$.



Grid: búsqueda

- Cada rectángulo puede ser visto como un bucket de una tabla de hashing. Luego, cada punto que pertenece al rectángulo en un registro en el bucket. Si es necesario se pueden considerar buckets de overflow.
- La estrucura de datos utilizada es un array n-dimensional de buckets.
- Para encontrar el bucket que contiene un determinado registro es necesario conocer los valores asociados a las líneas que generan las particiones.
- Más que una operación de hashing lo que se realiza es una búsqueda simple tomando cada componente del punto.

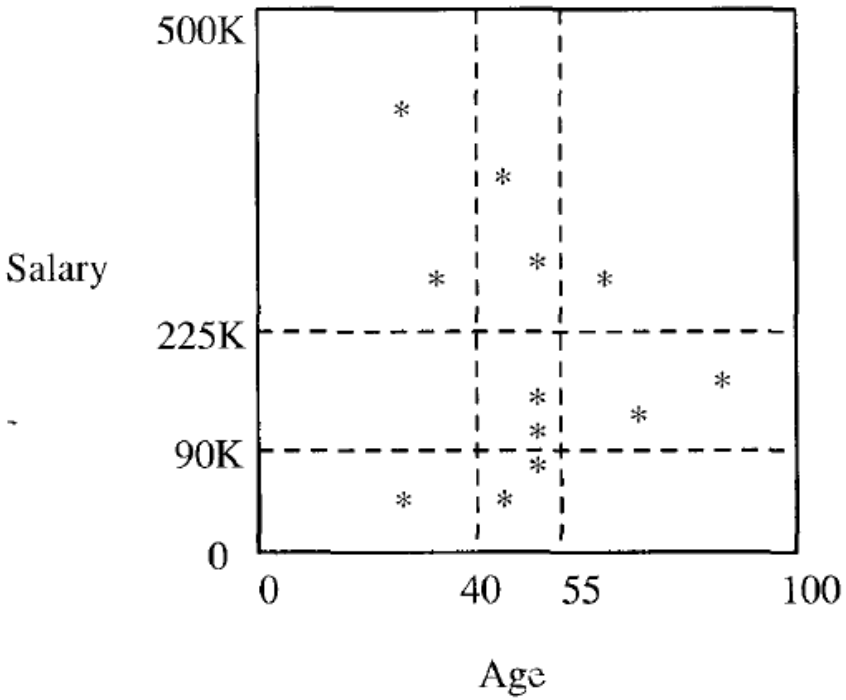
Índices Multidimensionales

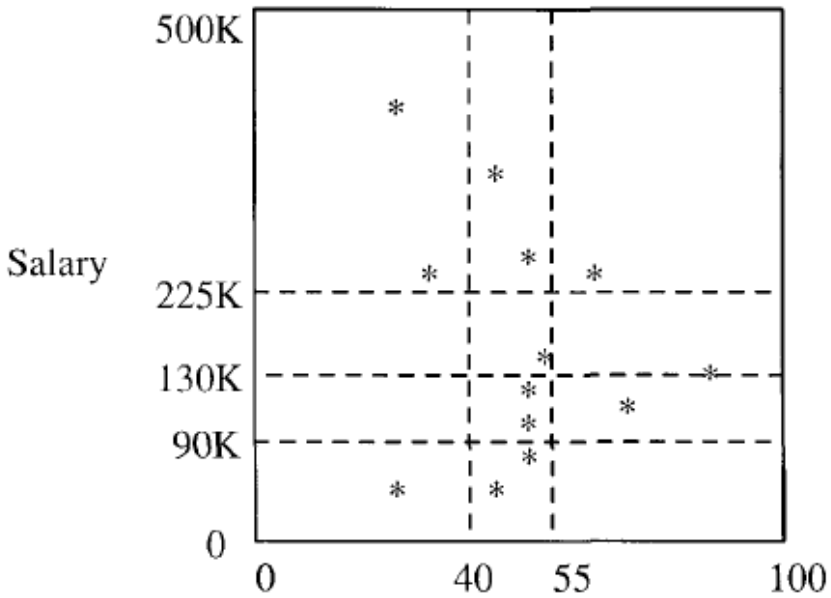


Grid: inserción

- Si hay espacio en el bucket no hay problema.
- En caso contrario: buckets de overflow o reparticionamiento.
- El reparticionamiento se logra moviendo/agregando líneas.

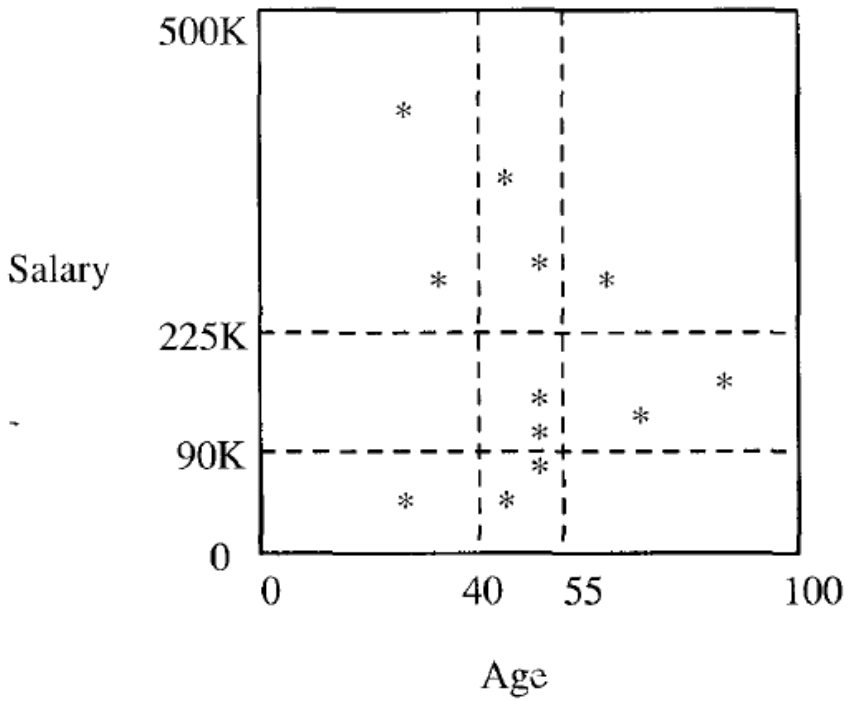
Ejemplo: Inserción del registro $(52, 200)$. Alternativas: (1) agregar línea vertical $edad = 51$. (2) agregar línea horizontal $salario = 130$ (3) agregar línea horizontal $salario = 100$.





Grid: rendimiento

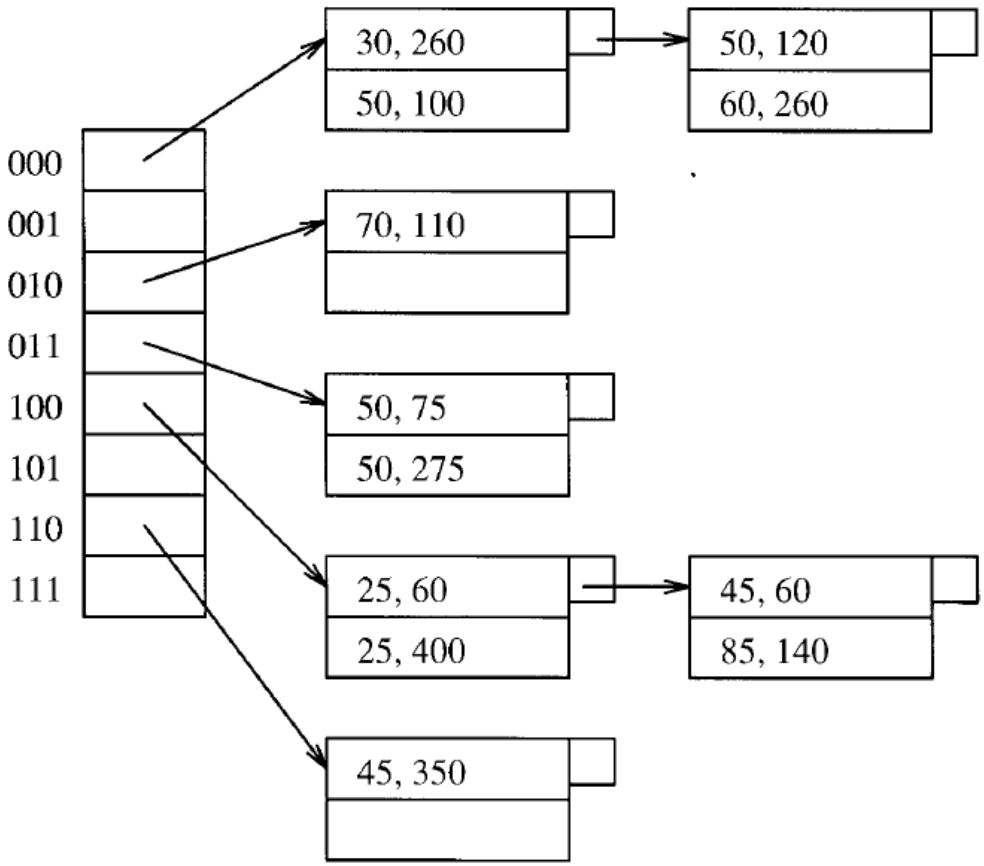
- buscar un registro específico
- encontrar todos los registros $edad = 50$.
- consultas por rango.
- nearest-neighbor



Partitioned hash functions

- En este tipo de índice el bucket se obtiene concatenando secuencias de bits generadas por n funciones de hashing $(h_1, h_2, ..., h_n)$. De este modo un registro con n atributos y valores $(v_1, v_2, ..., v_n)$ será almacenado en el bucket $h_1(v_1)h_2(v_2)...h_n(v_n)$
- Supongamos un tabla de hashing con buckets numerados utilizando 10 bits (1024 buckets), donde:
  - Los registros poseen atributos (a, b).
  - La función h_a se utiliza para obtener los 4 bits más significativos de la clave 0101.
  - La funcion h_b se utiliza para obtener los 64 bits restantes 111000.
  - El bucket donde se almacena (a, b) es 0101111000

Considerando el mismo conjunto de datos de índices grid. Se considera 3 bits (un bit es para la edad y los otros dos para el salario).
La función de hash para la edad es $edad\%2$ y para el salario $salario\%4$.
Una edad impar estará en un bucket de la forma $1b_2b_1$.



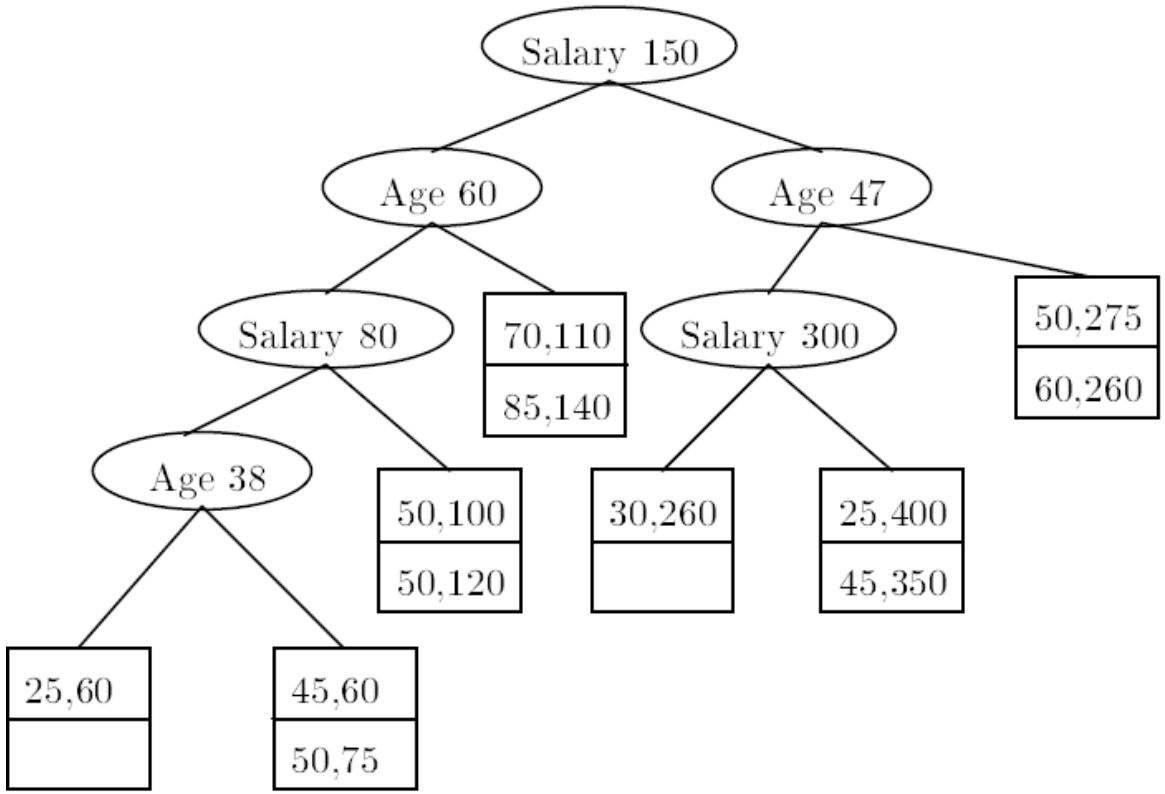
Grid v/s Partitioned hash functions

- Consultas del tipo rango o nearest-neighbor no pueden ser respondidas usando partitioned hash functions.
- Sin embargo, partitioned hash function es superior que Grid ante consultas del tipo "partial match".

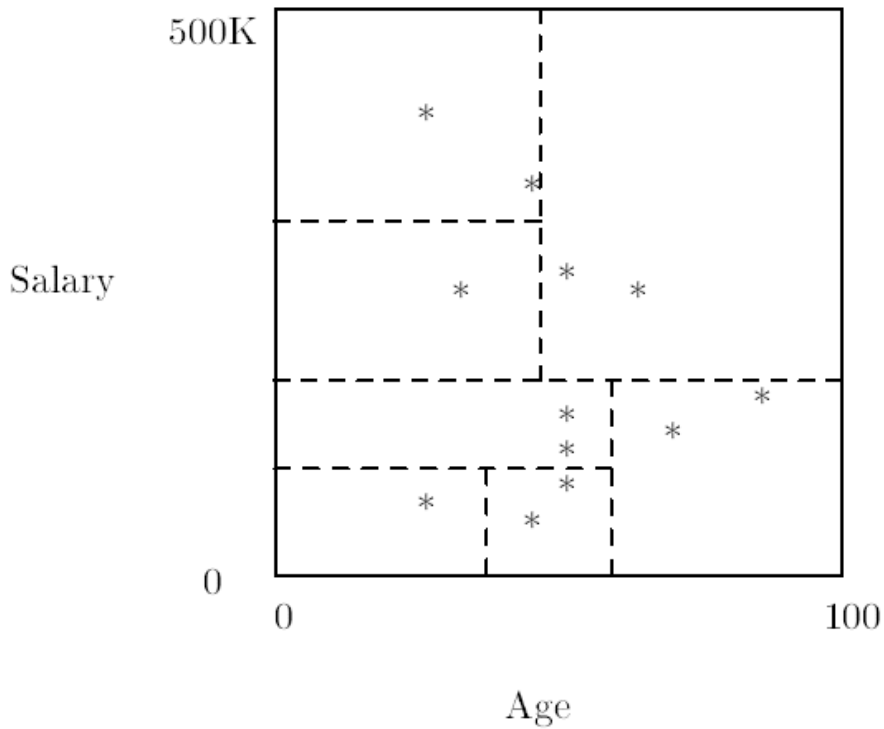
kd-tree

- Es una estructura de datos que generaliza un árbol de búsqueda binaria para datos multidimensionales.
- Es un árbol binario en el cual los nodos internos tienen asociado un atributo a y un valor V que divide el espacio de búsqueda en 2: una parte contiene registros (puntos) donde el valor de a es menor que V y la otra contiene puntos cuyo valor de a es mayor o igual que V.
- Diferentes niveles del árbol son particionados usando los distintos atributos en forma rotativa.
- Nodos
  - Interno: contienen sólo un atributo a, un valor V y punteros hacia sus hijos izq y der.
  - Hoja: cada nodo hoja es un bloque que contiene tantos registros como pueda almacenar.

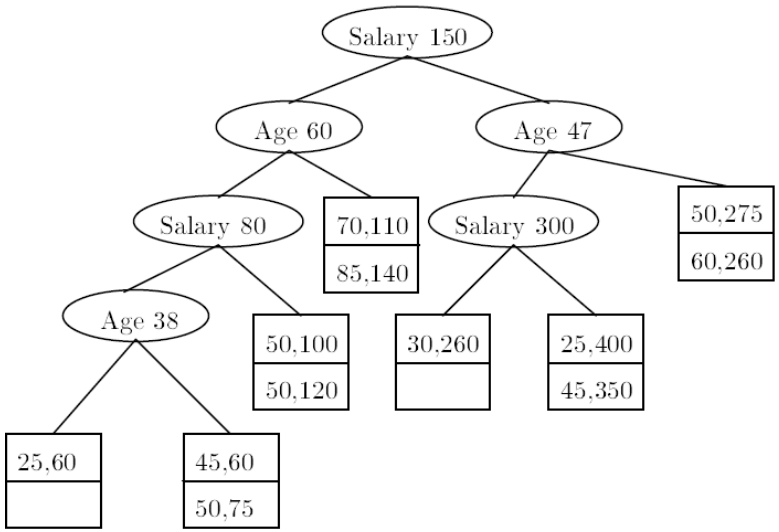
Ejemplo
Los siguientes puntos representan a un conjunto de clientes mediante los atributos $(edad, salario)$:
$(25, 60), (50, 120), (25, 400),$
$(45, 60), (70, 110), (45, 350),$
$(50, 75), (85, 140), (50, 275),$
$(50, 100), (30, 260), (60, 260)$

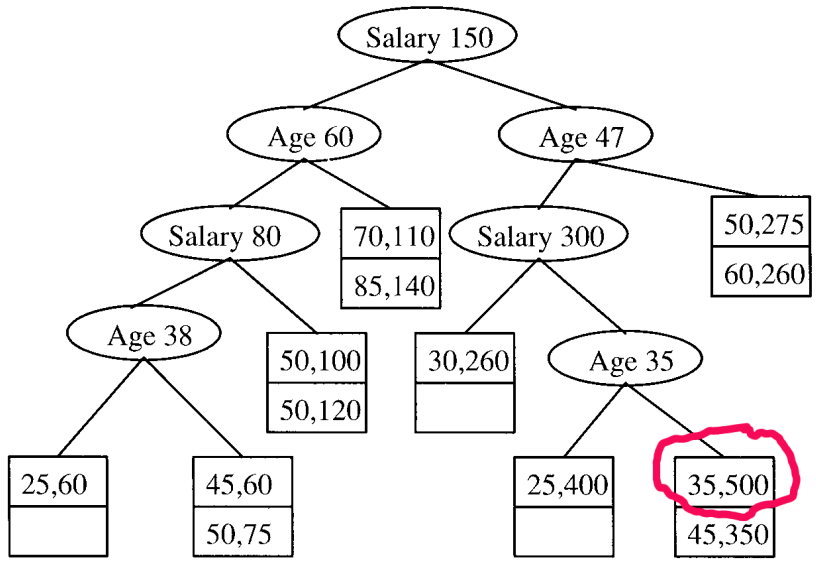


La figura muestra como los diferentes niveles del árbol dividen el espacio de búsqueda. La línea horizontal equivalente a $salario = 150$ representa la raíz. El espacio bajo esta línea corresponde al hijo izq y el espacio sobre la línea al hijo der.

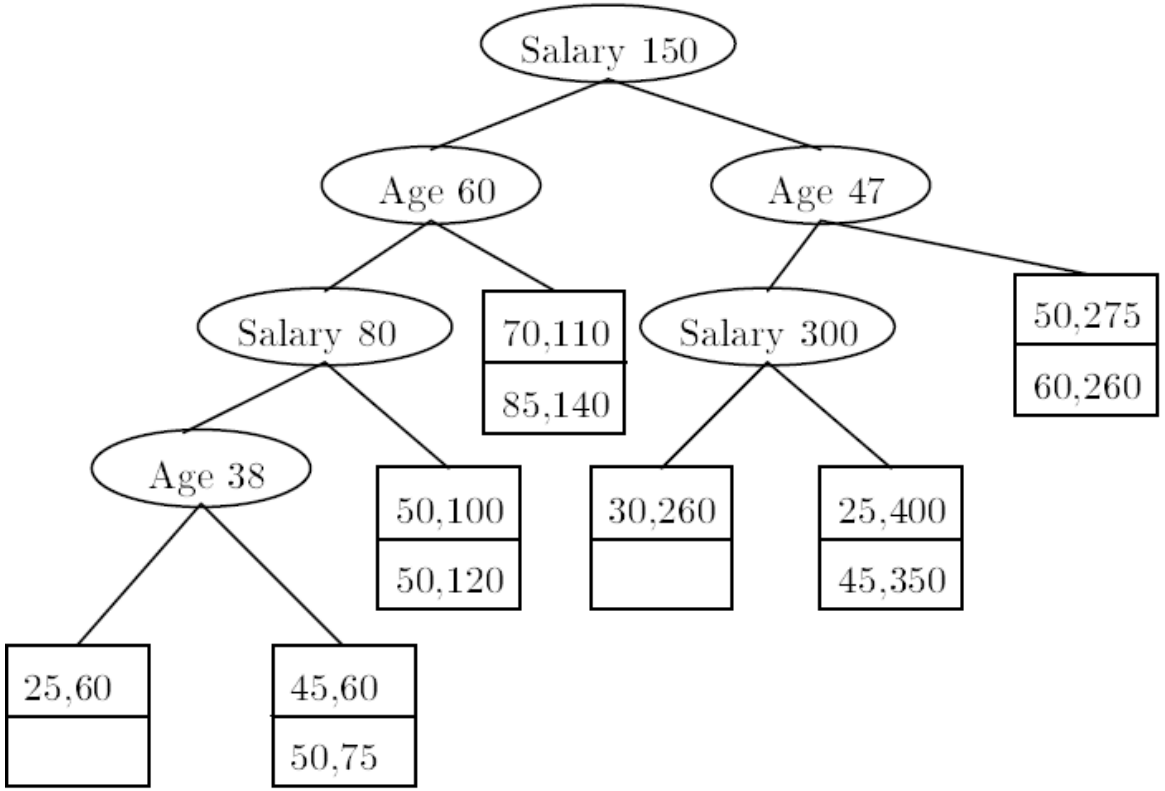


Inserción del registro $(35, 500)$. $500 > 150 \rightarrow 35 < 47 \rightarrow 500 > 150 \rightarrow$. Luego, el registro debe insertarse a la derecha del nodo $Salary = 300$. Sin embargo, ya que no hay espacio en el bloque se crea el nodo interno $Age = 35$ y se redistribuye el contenido de los bloques.





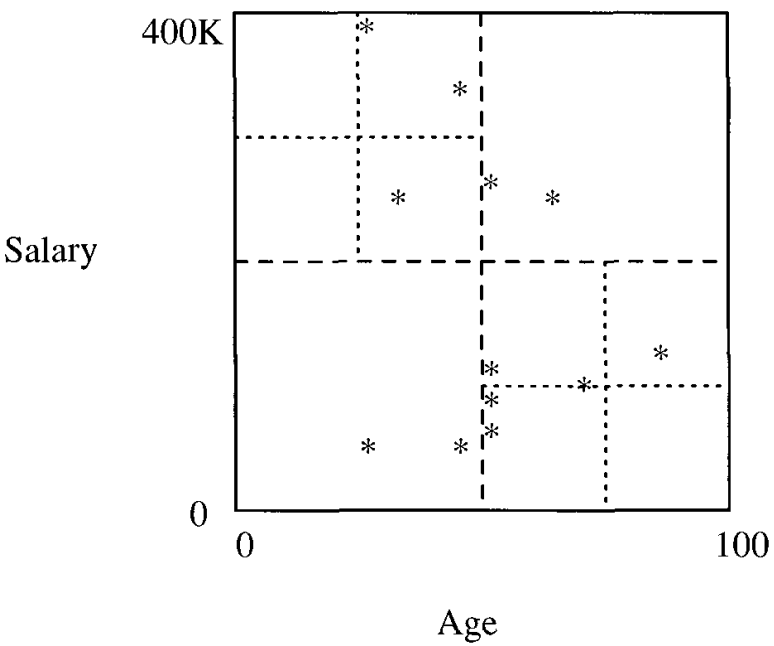
Cómo recorrer el árbol para responder la consulta: $35 \leq Age \leq 55$, $100 \leq Salary \leq 200$?

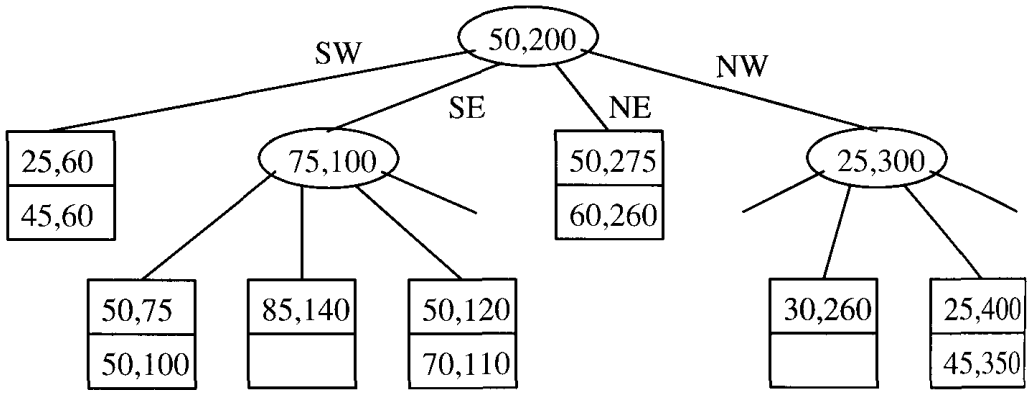


Quad tree

- En un quad tree cada nodo interno tiene 4 (2^d) hijos.
- Un nodo interno representa una región cuadrada.
- Si el nro de puntos dentro de una región $\leq fb$ entonces dicha región es una hoja. En caso contrario se genera un nodo interno con hijos que corresponden a los 4 cuadrantes.
- Algoritmo de inserción similar al de kd-tree. Sin embargo, cuando d es grande se pueden generar muchas regiones vacías.

Considerando $fb = 2$ y el mismo conjunto de datos de los ejemplos anteriores. Cada nodo interno indica el centro de su región.





Árboles R

- Son utilizados para organizar un conjunto de datos d-dimensionales representados por su MBR (minimum bounding rectangle).
- Nodo interno. posee entradas de la forma (mbr, p), donde p es un puntero a un nodo hijo y mbr es el MBR que contiene los MBRs del hijo.
- Nodo hoja. posee entradas de la forma (mbr, oid), donde mbr es el MBR que contiene al registro y oid es un identificador del registro.
- El mínimo de entradas en la raíz es 2.
- Todas las hojas del árbol se encuentran al mismo nivel.
- Un árbol R de orden $(m.M)$ es un árbol con un mínino de m y un máximo de M entradas por nodo.

Índices Multidimensionales