CẤU TRÚC DỮ LIỆU ĐA CHIỀU

Hệ cơ sở dữ liệu đa phương tiện

HK1, 2023 - 2024

Giới thiệu

Hai nhóm dữ liệu chính:

Dữ liệu điểm (Point data)

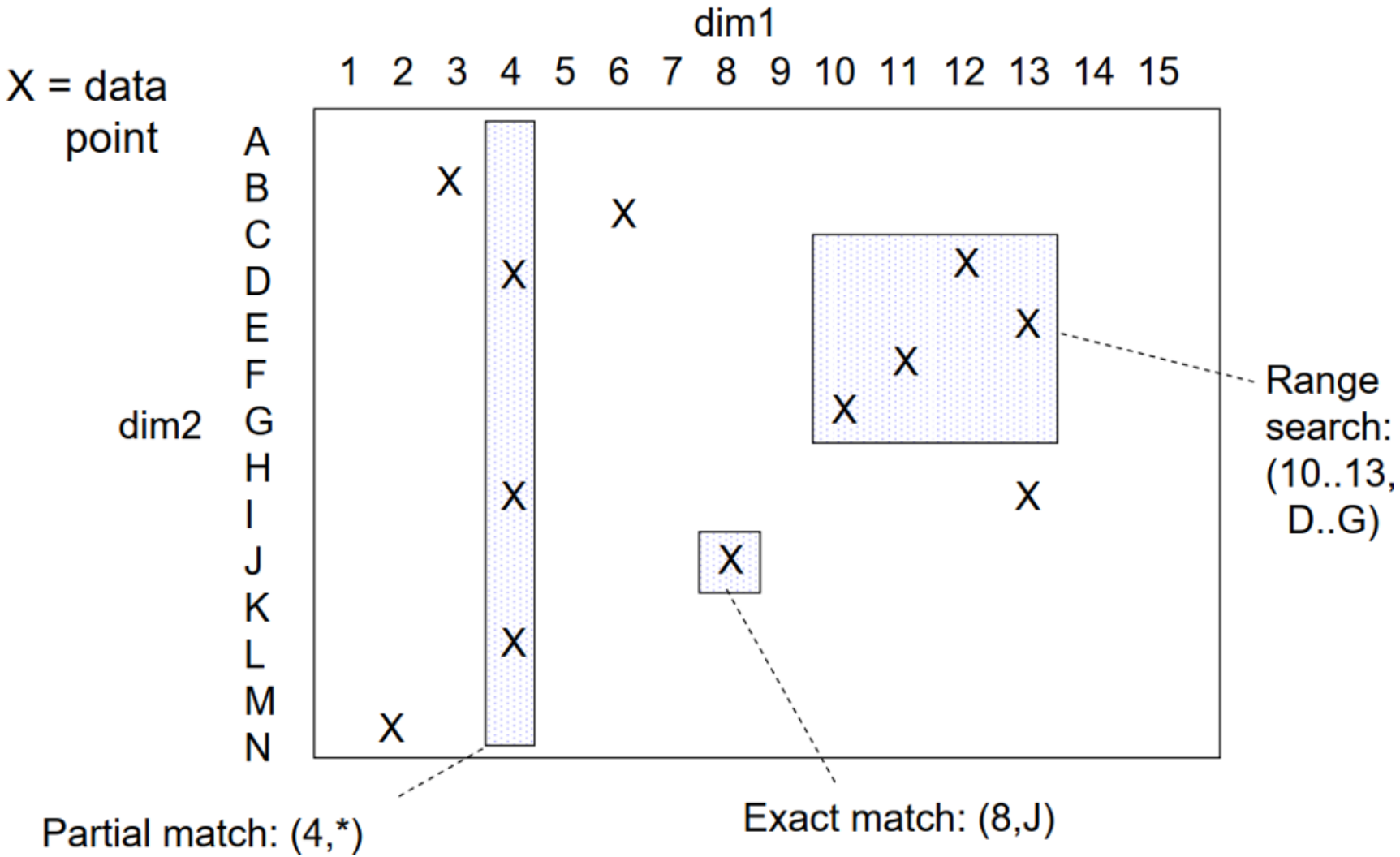
- Đối tượng csdl gồm k-tuple (bộ dữ liệu) trong không gian k chiều
- Mỗi phần tử bộ dữ liệu tương ứng với tọa độ trong không gian
- Truy xuất đa thuộc tính từ CSDL quan hệ, CSDL văn bản, vector đặc trưng của các đối tượng đa phương tiện

Dữ liệu không gian (Spatial data)

- Các đối tượng CSDL có một số loại hình dạng (shape) và kích thước (size) khác nhau
- Điểm là trường hợp đặc biệt
- Bản vẽ CAD, thiết kế VLSI, địa lý, xử lý ảnh...

Giới thiệu

Các loại truy vấn trong không gian 02 chiều



Các loại truy vấn đa chiều

Truy vấn đối sánh chính xác (exact-match queries)

- Tất cả các tọa độ (thuộc tính) được cố định trong truy vấn. Độ phức tạp là hàm log

Truy vấn đối sánh một phần (partial-match queries)

- Chỉ t trong k tọa độ được xác định trong truy vấn. Phần còn lại có giá trị tùy ý
- Cận dưới cho độ phức tạp trong trường hợp xấu nhất $O(n^{1-\frac{t}{k}})$

Các loại truy vấn đa chiều

Truy vấn phạm vi (range queries)

- Đối với mỗi chiều, một phạm vi giá trị được xác định
- Đối sánh chính xác: phạm vi $= [-c, c]$; đối sánh một phần $= [-\infty, \infty]$ đối với mỗi tọa độ

Truy vấn phù hợp nhất (best-match queries)

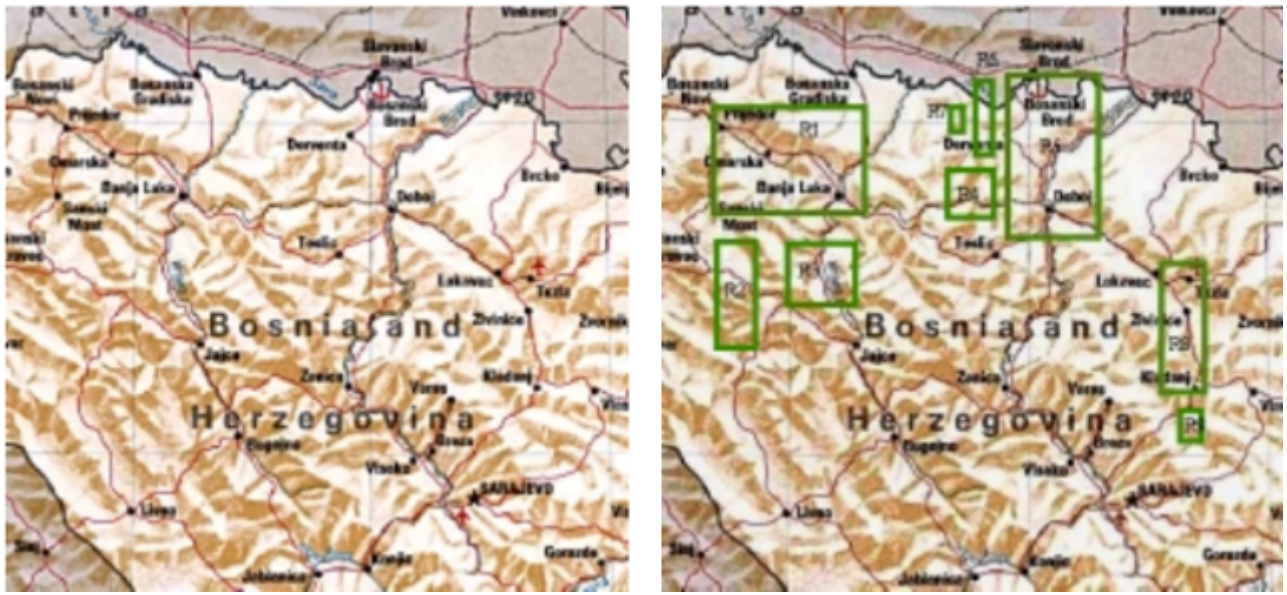
- Tìm láng giềng gần nhất của điểm/khu vực được chỉ định bởi điều kiện truy vấn (chính xác hoặc phạm vi)

Tìm k láng giềng gần nhất (Finding k nearest neighbors)

Truy vấn xếp hạng (Ranking query)

Cấu trúc dữ liệu đa phương tiện

- Bản đồ có thể xem như hình ảnh 2 chiều
  - Point: là các vị trí cần quan tâm
- Các điểm được lưu trữ trong cấu trúc dữ liệu chuyên biệt: k-d Tree, Point quadtree, MX-Quadtree, R-trees

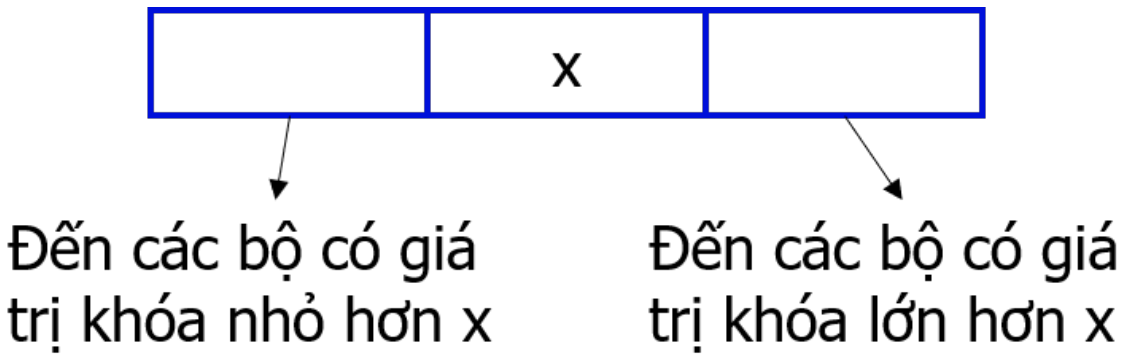


k-D trees

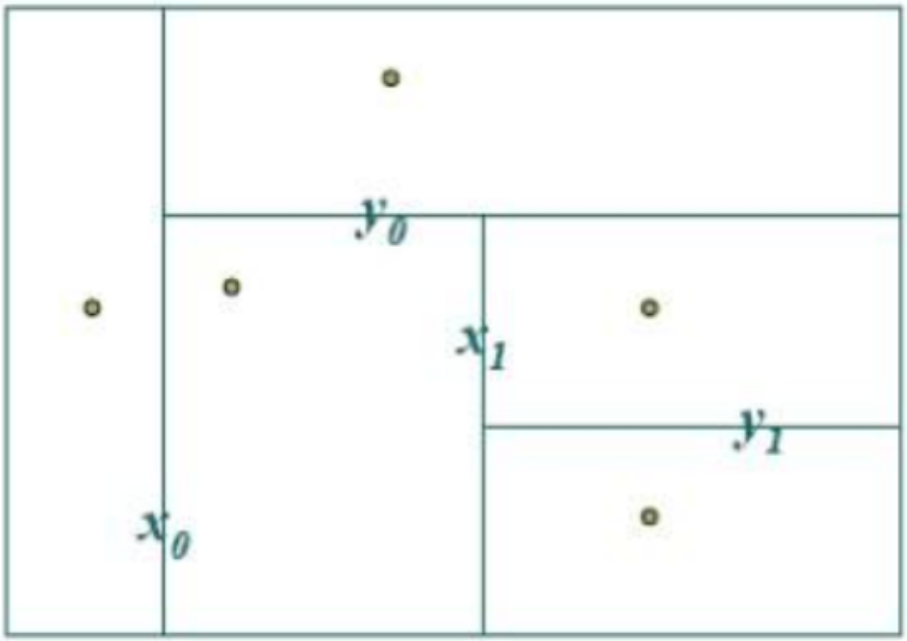
- Dành lưu trữ dữ liệu điểm đa chiều (k-dimensional point)
  - 2-tree: lưu dữ liệu điểm 02 chiều
  - 3-tree: lưu dữ liệu điểm 03 chiều
  - ...
  - Mỗi điểm là vector có k phần tử
- Không lưu dữ liệu không gian (dữ liệu vùng)

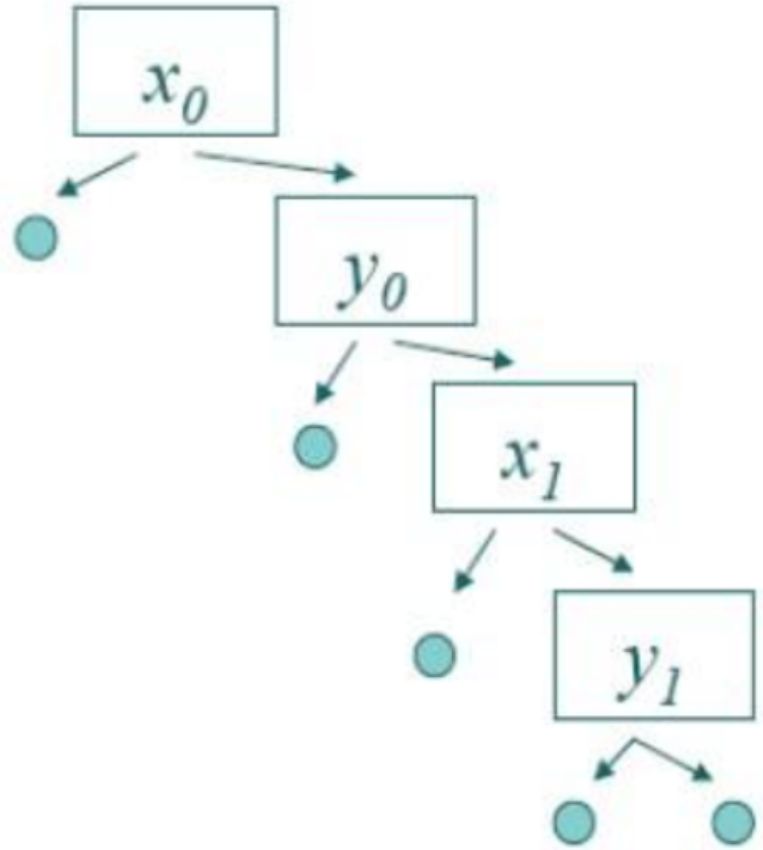
k-D trees

- Là mở rộng của cây nhị phân
- Ở mỗi mức, các bản ghi sẽ được chia theo giá trị của 01 chiều nhất định
  - Mức 0: giá trị chiều 0
  - ...
  - Mức $k-1$: giá trị chiều $k-1$
  - Mức k: giá trị chiều 0



2-D trees





k-D trees

- Cấu trúc 1 nút

INFO	XVAL	YVAL
LLINK	RLINK	

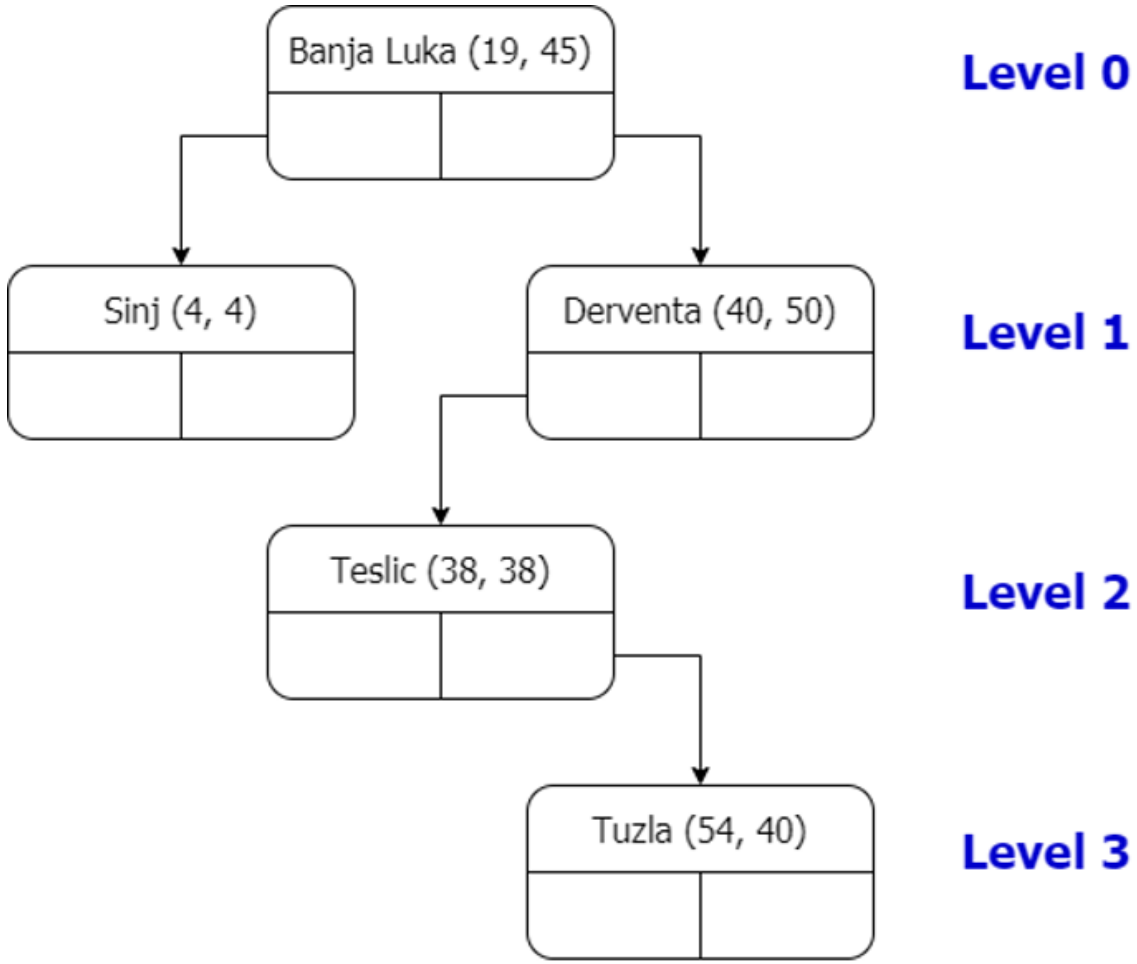
Định nghĩa: 2-D tree là cây nhị phân thỏa mãn

- Nếu nút N ở mức chẵn:
  $(\forall M \in N.\mathrm{LLINK} : M.\mathrm{XVAL} < N.\mathrm{XVAL}) \quad \wedge$
  $(\forall P \in N.\mathrm{RLINK} : P.\mathrm{XVAL} \geq N.\mathrm{XVAL})$
- Nếu nút N ở mức lẻ:
  $(\forall M \in N.\mathrm{LLINK} : M.\mathrm{YVAL} < N.\mathrm{YVAL}) \quad \wedge$
  $(\forall P \in N.\mathrm{RLINK} : P.\mathrm{YVAL} \geq N.\mathrm{YVAL})$

k-D trees

Ví dụ: Thứ tự insert

INFO	XVAL	YVAL
Banja Luka	19	45
Derventa	40	50
Teslic	38	38
Tuzla	54	40
Sinj	4	4



k-D trees

Chèn (insert)/Tìm kiếm (search)

- Nút cần thêm: $P(\text{info}, x, y)$
- Các bước (lặp lại cho đến khi kết thúc)
  - Nút đang duyệt: N
  - Nếu N.XVAL $= x$ và N.YVAL $= y$ thì ghi đè N và kết thúc
  - Nếu N ở *level chẵn* (0, 2, . . .)
    - Nếu $x < N$.XVAL thì duyệt cây con bên trái
    - Ngược lại, duyệt cây con bên phải
  - Nếu N ở *level lẻ* (1, 3, . . .)
    - Nếu $y < N$.YVAL thì duyệt cây con bên trái
    - Ngược lại, duyệt cây con bên phải

k-D trees

Xóa (delete) trong 2-D tree

Gọi T là một 2-D tree, nút cần xóa có $\mathrm{XVAL} = x$ và $\mathrm{YVAL} = y$

- Tìm N: $N.\mathrm{XVAL} = x$ và $N.\mathrm{YVAL} = y$
- Nếu N là ***nút lá***: đặt LLINK/RLINK của cha N về NIL, giải phóng $N \rightarrow$ kết thúc
- Nếu N là ***nút trong***
  - Tìm nút thay thế R ở trong 2 cây con T_l và T_r
  - Thay các giá trị không phải con trỏ bằng giá trị của R
  - Lặp để xóa R

k-D trees

Xóa (delete) trong 2-D tree

Gọi T là một 2-D tree, nút cần xóa $P(\text{info}, x, y)$

- Tìm nút thay thế cho nút cần xóa:

  mọi nút thuộc cây con trái/phải của N cũng thuộc cây con trái/phải tương ứng của R

  - Nếu nút N ở mức chẵn
    $(\forall M \in N.\text{LLINK} : M.\text{XVAL} < R.\text{XVAL}) \quad \wedge$
    $(\forall P \in N.\text{RLINK} : P.\text{XVAL} \geq R.\text{XVAL})$
  - Nếu nút N ở mức lẻ
    $(\forall M \in N.\text{RLINK} : M.\text{YVAL} < R.\text{YVAL}) \quad \wedge$
    $(\forall P \in N.\text{RLINK} : P.\text{YVAL} \geq R.\text{YVAL})$

k-D trees

Xóa (delete) trong 2-D tree

Gọi T là một 2-D tree, nút cần xóa $P(\text{info}, x, y)$

Tìm nút thay thế

- Nếu N: level chẵn
  - T_r: không rỗng
    $\rightarrow$ Nút R trong cây con T_r có giá trị XVAL nhỏ nhất sẽ là nút thay thế
  - T_r: rỗng
    - Tìm nút thay thế bên cây T_l: R' bên cây T_l có XVAL nhỏ nhất
    - $N.\text{RLINK} = N.\text{LLINK}$, $N.\text{LLINK} = \text{NIL}$
- Nếu N: level lẻ $\rightarrow$ thực hiện tương tự với giá trị YVAL

k-D trees

Truy vấn (query) trên phạm vi 2-D tree

- Truy vấn phạm vi: 1 điểm (x_c, y_c) và 1 khoảng cách r
- Tìm các điểm (x, y) trên cây 2-D sao cho khoảng cách đến $(x_c, y_c) \leq r$

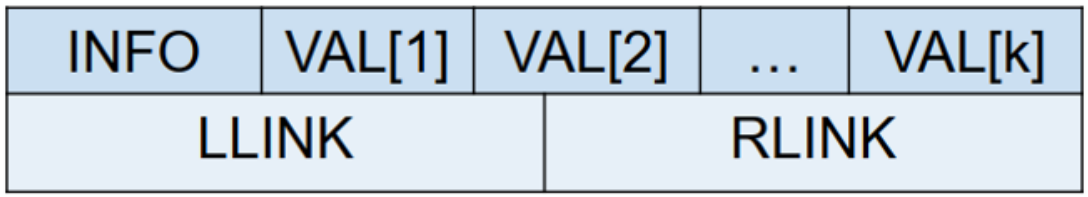
Các dạng khoảng cách

- L1:
  $x_c - r \leq x \leq x_c + r \qquad y_c - r \leq y \leq y_c + r$
- L2 (Euclidean)
- ...

k-D trees

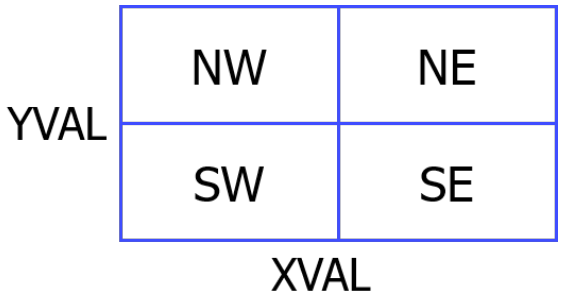
- $P(x_1, x_2, \ldots, x_k)$
- N: 1 nút thuộc k-D tree nếu
  - Với mọi nút M thuộc cây bên trái của N: $M.\text{VAL}[i] < N.\text{VAL}[i]$
  - Với mọi nút P thuộc cây bên phải của N: $P.\text{VAL}[i] \geq N.\text{VAL}[i]$

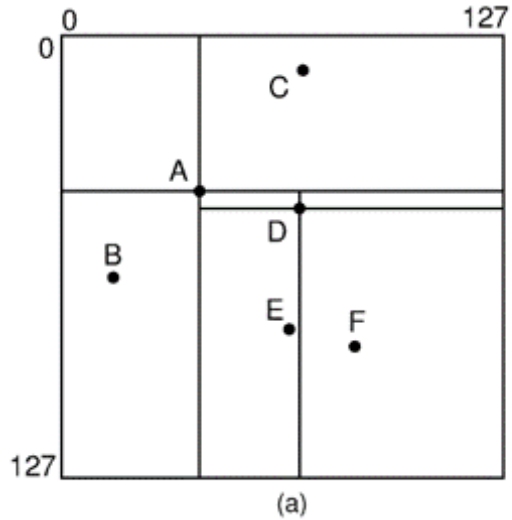
với $i = \text{level}(N) \mod k$

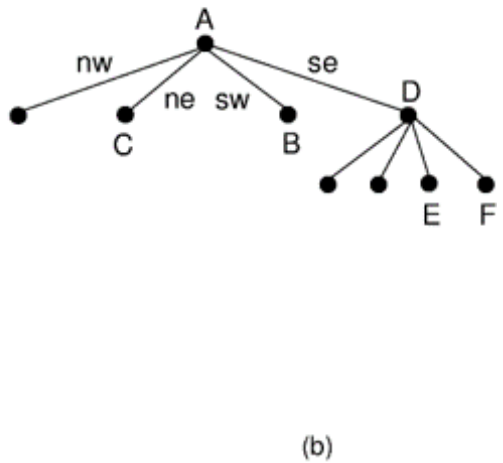


Cây tứ phân dạng điểm (Point Quadtrees)

- Mỗi điểm trong cây sẽ chia 1 vùng thành 4 vùng con theo cả 2 chiều ngang và dọc (N.XVAL và N.YVAL): NW (Northwest), SW (Southwest), NE (Northeast), SE (Southeast)

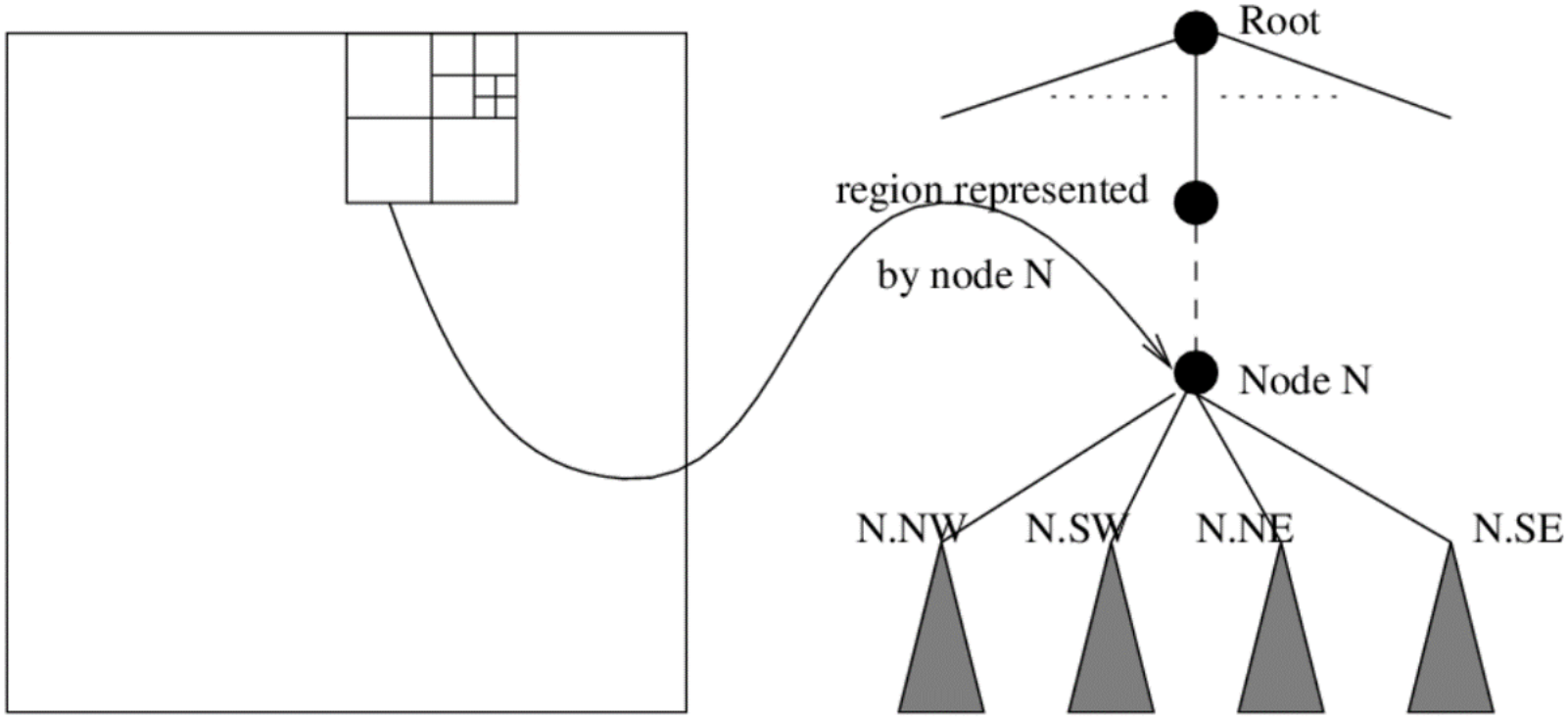






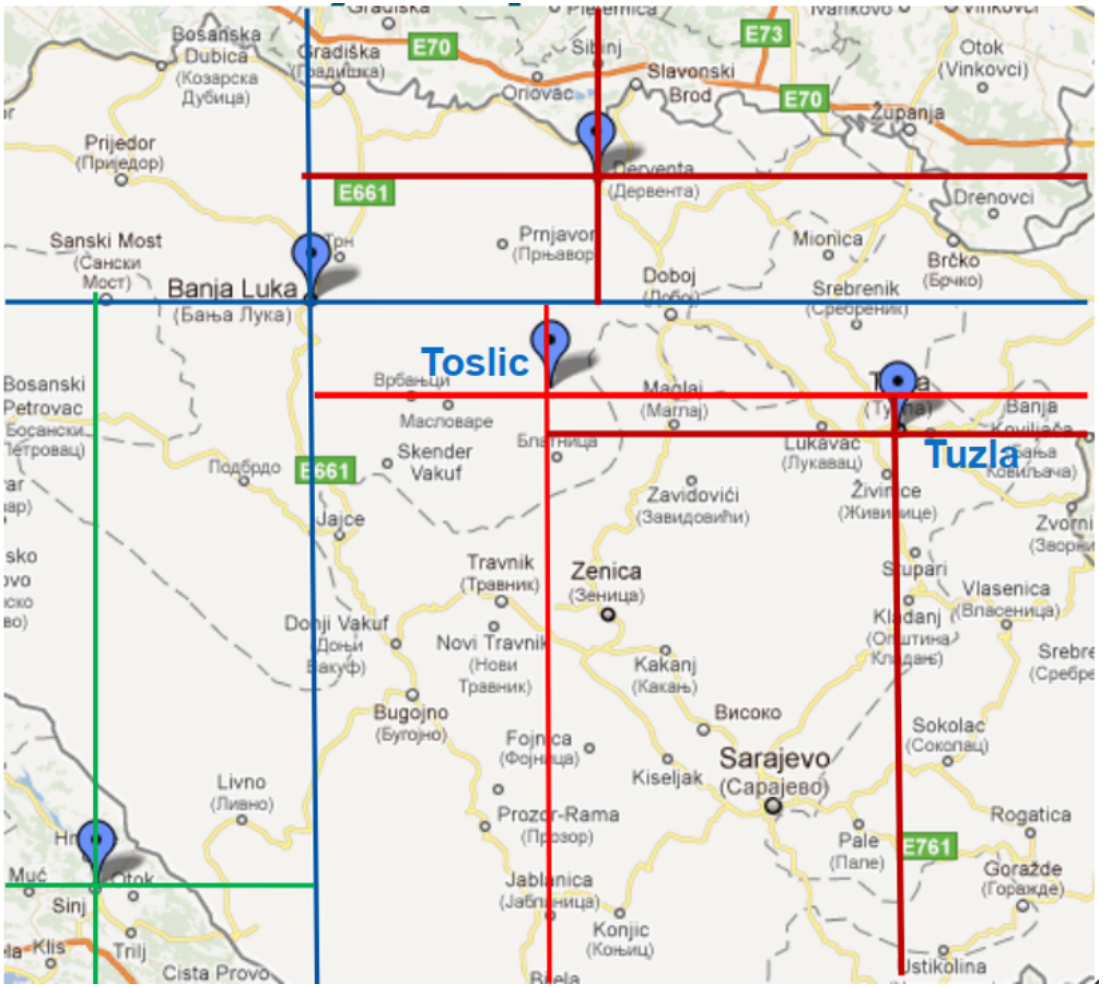
Cây tứ phân dạng điểm (Point Quadtrees)

Mỗi nút trong cây tứ phân ngầm biểu diễn 1 vùng



Cây tứ phân dạng điểm (Point Quadtrees)

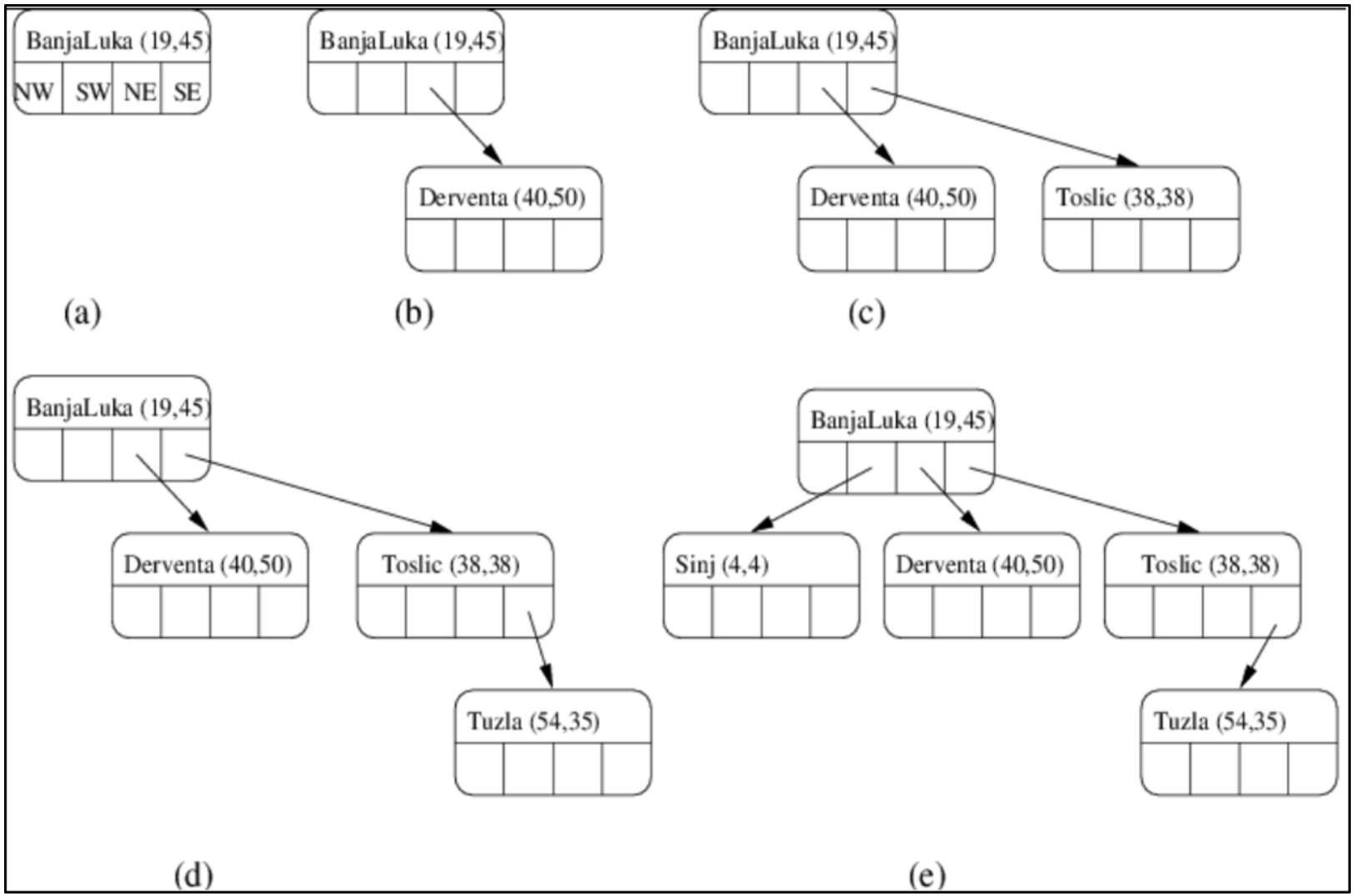
Thêm dữ liệu vào cây tứ phân



- Banja Luka (19, 45)
- Derventa (40, 50)
- Toslic (38, 38)
- Tuzla (54, 40)
- Sinj (4, 4)

Cây tứ phân dạng điểm (Point Quadtrees)

Thêm dữ liệu vào cây tứ phân



Cây tứ phân dạng điểm (Point Quadtrees)

Xóa dữ liệu trong cây tứ phân

- Nút lá: đơn giản

$\rightarrow$ Thiết lập lại con trỏ ở nút cha = NIL và giải phóng nút cần xóa

- Nút trong: phức tạp

$\rightarrow$ Cần tìm nút thay thế

Ví dụ: xóa nút gốc

Cây tứ phân dạng điểm (Point Quadtrees)

Tìm nút thay thế khi xóa nút trong

Nút xóa là nút N

- Nút thay thế R đảm bảo
  $\forall R_1 \in N.\mathrm{NW} \rightarrow R_1 \in R.\mathrm{NW}$
  $\forall R_2 \in N.\mathrm{SW} \rightarrow R_2 \in R.\mathrm{SW}$
  $\forall R_3 \in N.\mathrm{NE} \rightarrow R_3 \in R.\mathrm{NE}$
  $\forall R_4 \in N.\mathrm{SE} \rightarrow R_4 \in R.\mathrm{SE}$
- Ví dụ: N: “Banja Luka” $\Rightarrow$ R: “Toslic”
- Không phải lúc nào cũng tìm được nút thay thế

Cây tứ phân dạng điểm (Point Quadtrees)

Xóa nút trong $\rightarrow$ tìm nút thay thế

$\rightarrow$ chèn lại các nút trỏ bởi NW, SW, NE, SE

$\rightarrow$ Trường hợp xấu nhất: tất cả các nút bị thay đổi

Cây tứ phân dạng điểm (Point Quadtrees)

Truy vấn trên cây tứ phân

procedure RangeQueryPointQuadtree(T: newqtnodetype, C: circle)

if region(T) $\cap$ C $= \emptyset$ **then** Halt

if (T.XVAL, T.YVAL) $\in$ C **then** print(T.XVAL, T.YVAL)

else

RangeQueryPointQuadtree(T.NW, C)

RangeQueryPointQuadtree(T.SW, C)

RangeQueryPointQuadtree(T.NE, C)

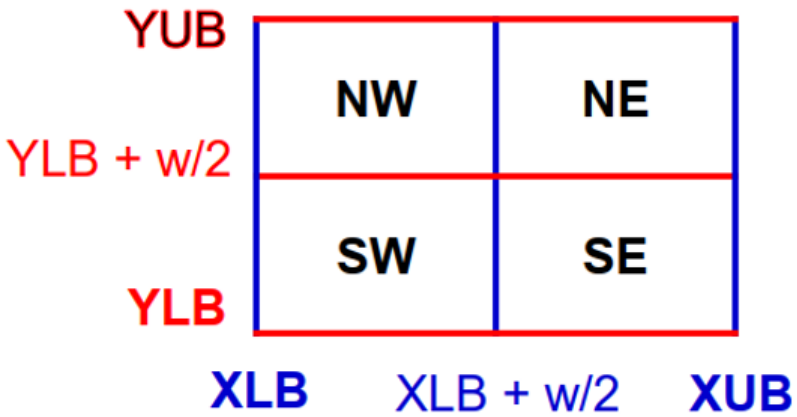
RangeQueryPointQuadtree(T.SE, C)

MX-Quadtrees

- Hình dáng cây không phụ thuộc vào
  - Số nút thêm vào
  - Thứ tự thêm vào
- Cho phép xóa và truy vấn hiệu quả
- Dữ liệu được chia theo lưới $2^k \times 2^k$
- k tự chọn, nhưng sau khi chọn thì không được thay đổi
- Cấu trúc nút
  - Tương tự cây tứ phân dạng điểm
  - Thông tin về vùng biểu diễn (XLB, XUB, YLB, YUB)
  - Nút gốc (root): XLB $= 0$, XUB $= 2^k$, YLB $= 0$, YUB $= 2^k$

MX-Quadtrees

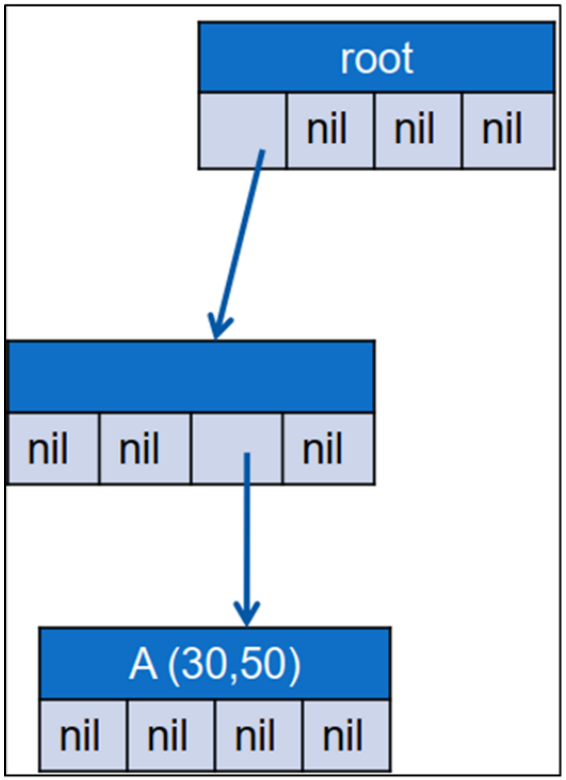
Các nút con của N (với w = N.XUB – N.XLB)

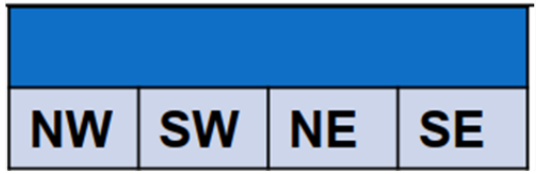


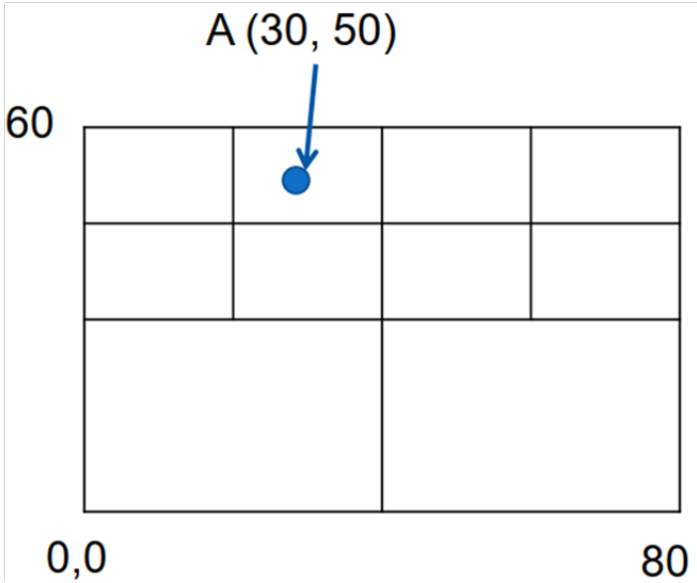
Child	XLB	XUB	YLB	YUB
NW	N.XLB	N.XLB+$\frac{w}{2}$	N.YLB+$\frac{w}{2}$	N.YLB+w
SW	N.XLB	N.XLB+$\frac{w}{2}$	N.YLB	N.YLB+$\frac{w}{2}$
NE	N.XLB+$\frac{w}{2}$	N.XLB+w	N.YLB+$\frac{w}{2}$	N.YLB+w
SE	N.XLB+$\frac{w}{2}$	N.XLB+w	N.YLB	N.YLB+$\frac{w}{2}$

MX-Quadtrees

Thêm (insert)

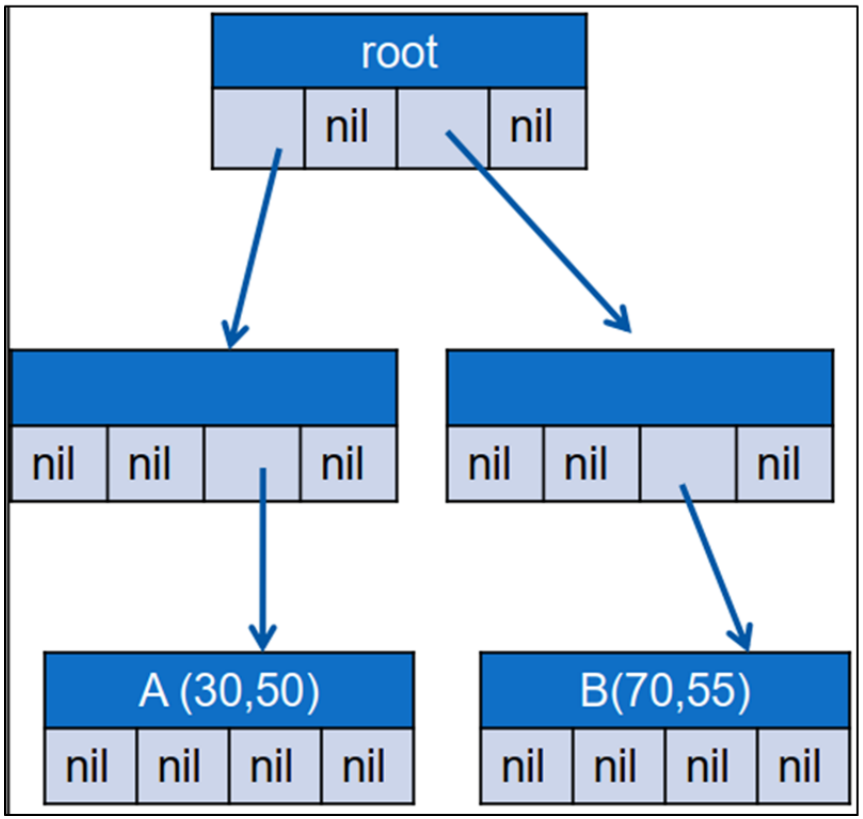


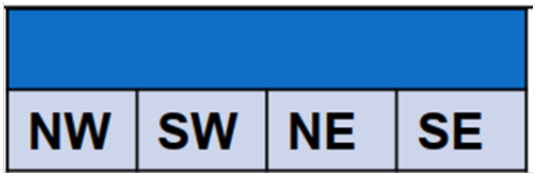


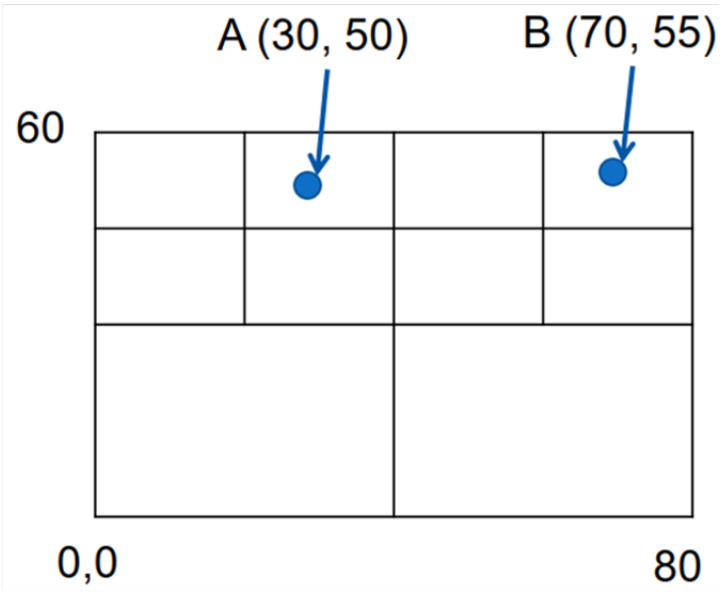


MX-Quadtrees

Thêm (insert)





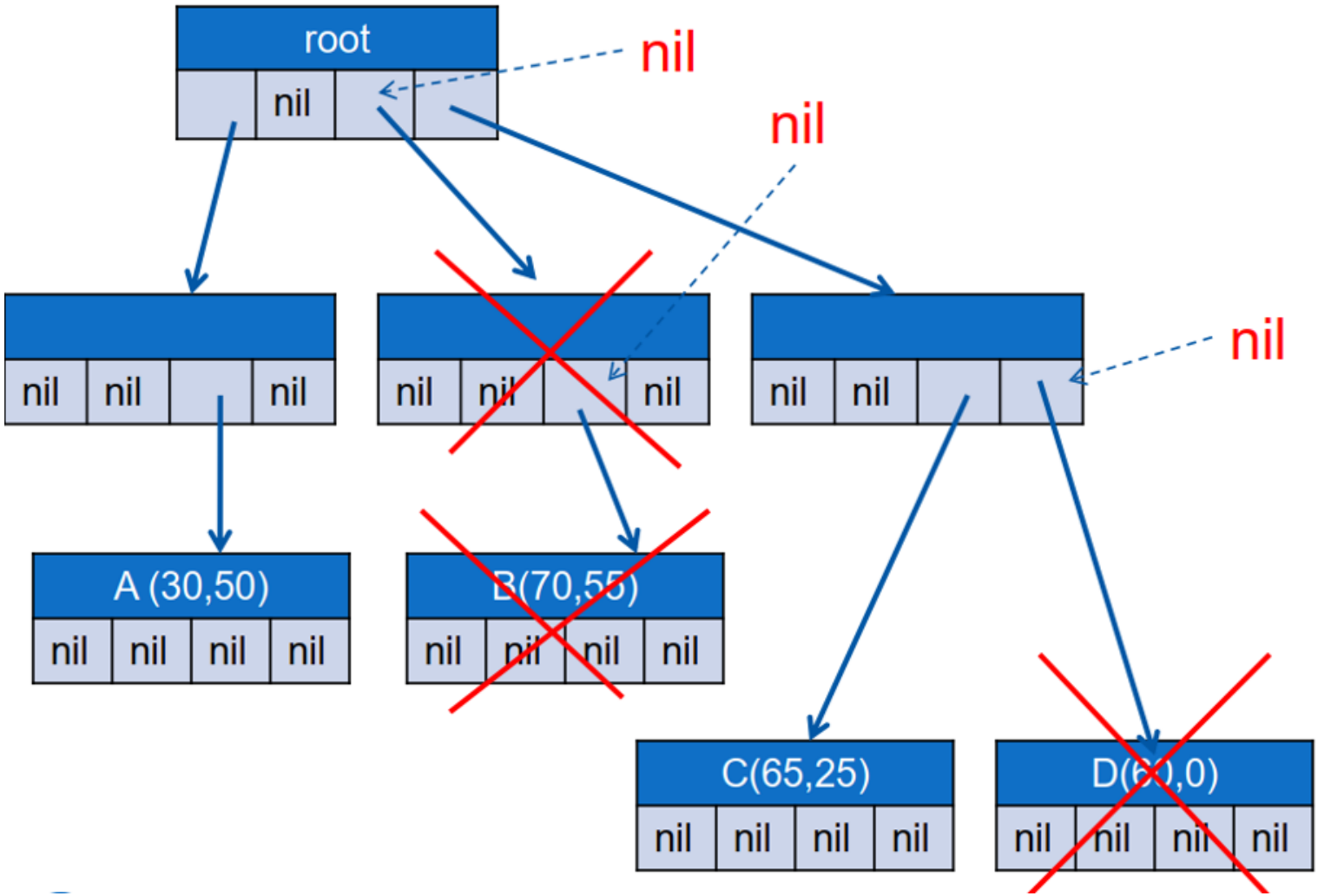


MX-Quadtrees

- Tất cả các điểm được biểu diễn bởi nút lá
- Nếu N là nút trong của cây MX-Quadtree, thì vùng biểu diễn bởi N chứa ít nhất 1 điểm dữ liệu
- Xóa dễ dàng với độ phức tạp O(k)

MX-Quadtrees

Xóa (delete)



MX-Quadtrees

- Tương tự cây tứ phân dạng điểm
- Điểm khác biệt: Kiểm tra 1 điểm nằm trong phạm vi truy vấn chỉ thực hiện ở mức lá
- **MX-Quadtree**
  - Dư nhiều nút nếu phân bố điểm thưa
  - Nếu nhiều hơn 01 có trong vùng nhỏ nhất
- **PR-Quadtree**: biến thể của MX-quadtree
  - Nếu 1 nút chỉ có 1 điểm $\rightarrow$ lưu vào nút đó mà không cần lưu vào nút lá
  - Chỉ phân chia vùng nếu vùng chứa nhiều hơn 1 điểm