UNIUNEA EUROPEANĂ

GUVERNUL ROMÂNIEI

Instrumente Structurale
2007-2013

Platformă de e-learning și curriculă e-content pentru învățământul superior tehnic

Geometrie computationala

23. Cautari in spatii ortogonale: Arbori de cautare d-dimensionali. Interogari

Arbori de cautare d-dimensionali

Ideea se generalizeaza direct la d dimensiuni:

- Se creaza o serie de arbori, unul pentru fiecare dimensiune.
- Se efectueaza cautarea ca mai inainte.

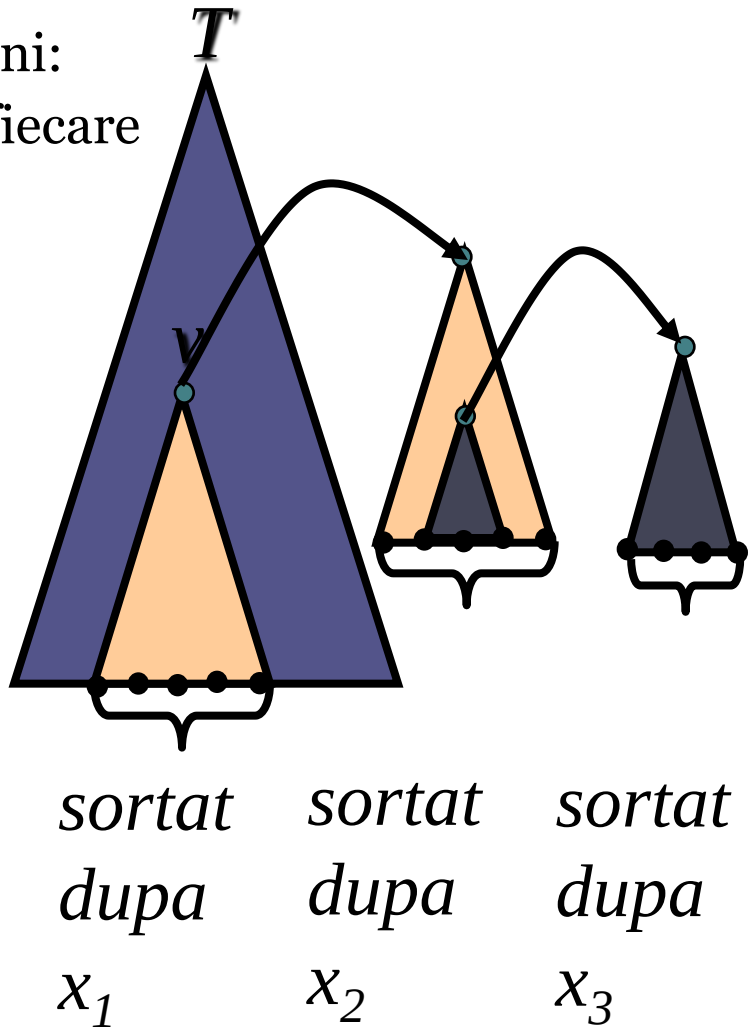
Complexitate:

- Constructie (timp si spatiu):

$$T_d(n) = O(n \lg n) + T_{d-1}(n) \times O(\lg n)$$
$$= O(n \lg^{d-1} n)$$

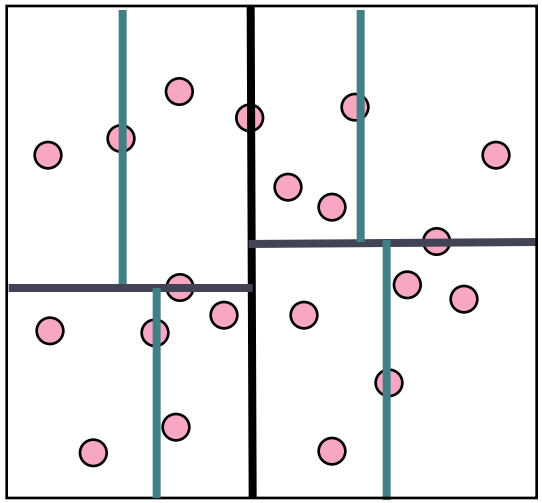
- Interogare:

$$Q_d(n) = O(\lg n) + Q_{d-1}(n) \times O(\lg n)$$
$$= O(\lg^d n)$$

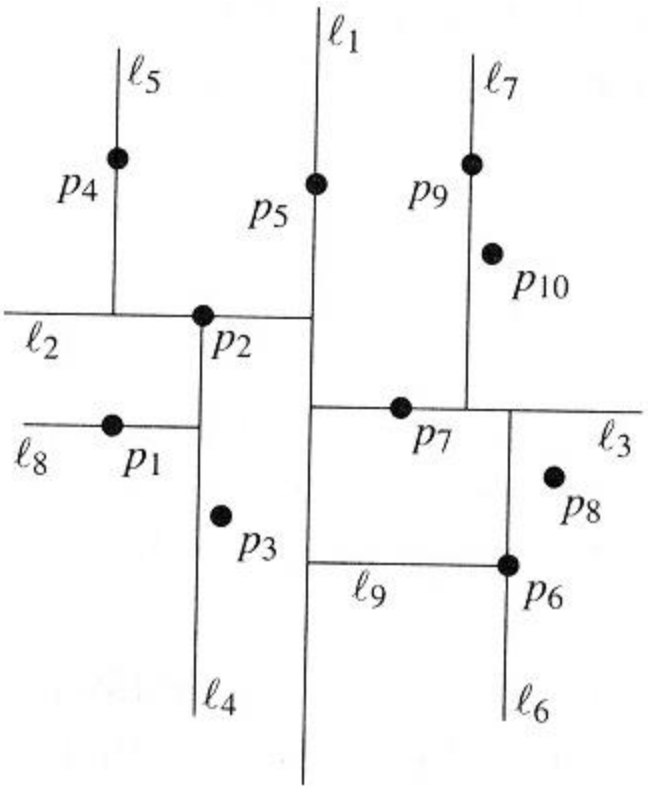


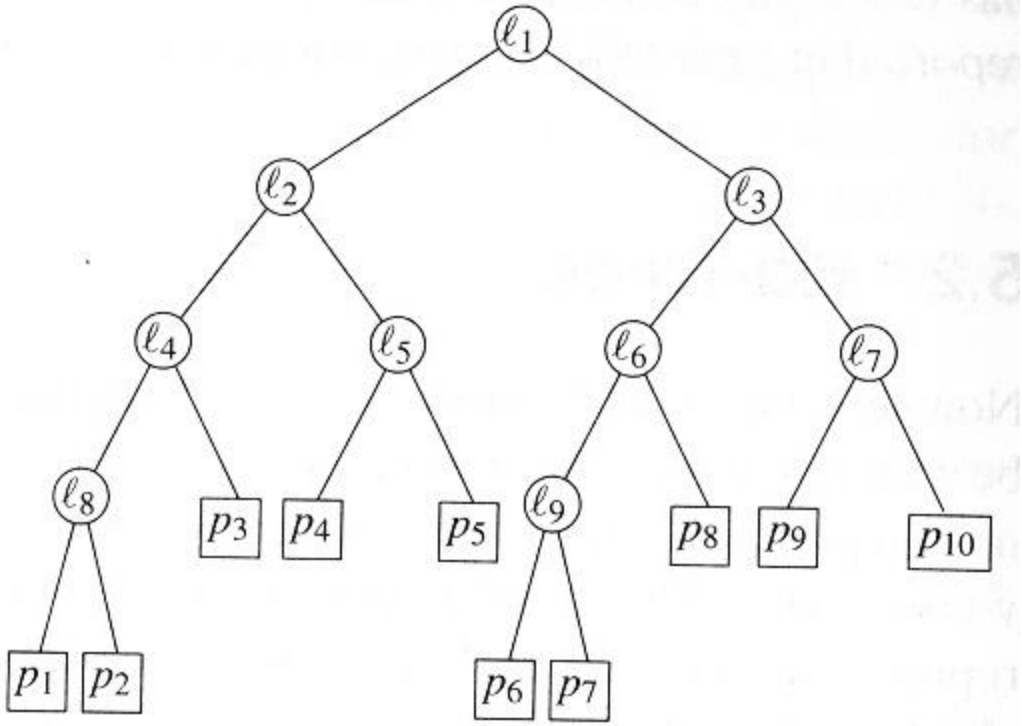
Arbori *kd* in 2D: idee

- Se incadreaza punctele cu un dreptunghi.
- Se impart punctele in doua submultimi de dimensiuni egale, printr-o linie orizontala sau verticala.
- Se continua partitionarea recursiv, alternand directia liniilor, pana cand submultimile sunt destul de mici (de dimensiuni constante).
- Submultimile canonice reprezinta subarbori.
- In dimensiuni superioare (k): Directiile de impartire alterneaza intre k axe (→ arbori *kd*).



Arbori *kd* in 2D: exemplu



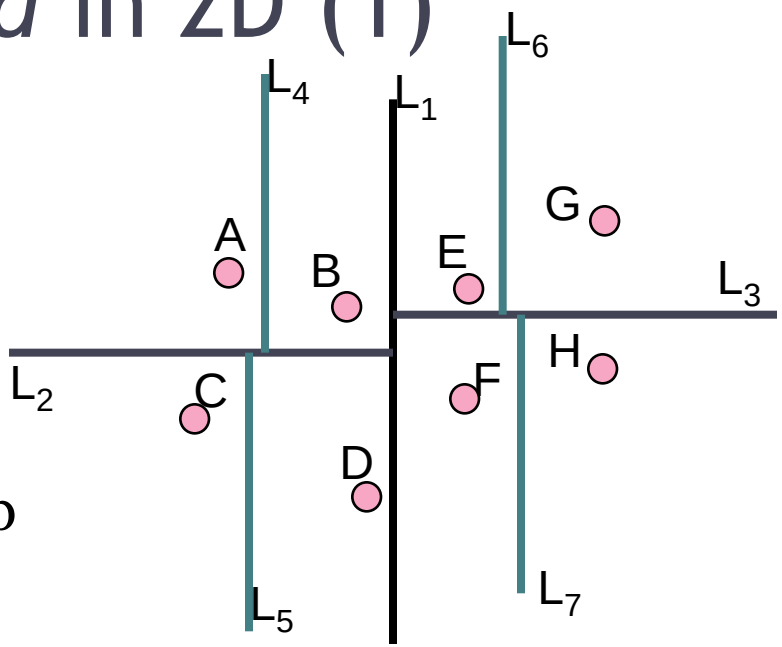


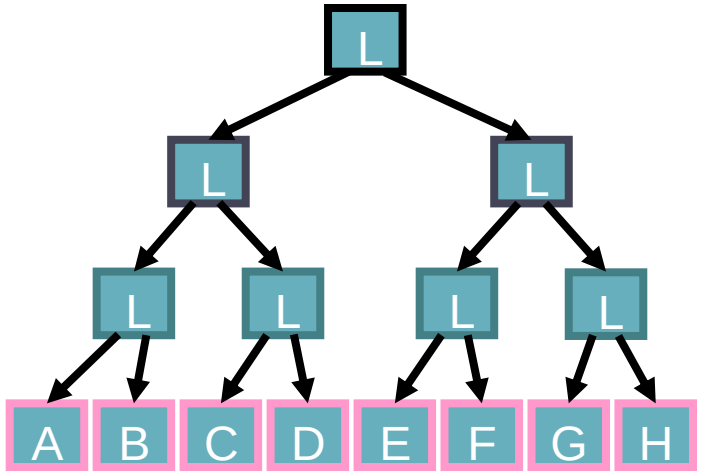
Construirea arborelui *kd* in 2D (1)

- Se partitioneaza planul in zone rectangulare aliniate la axe.
- Nodurile reprezinta linii de partitie, iar frunzele reprezinta puncte la intrare.
- Gasirea medianei necesita doar timp liniar!
- Complexitate de timp:

$$T(n) = \begin{cases} O(1) & n = 1 \\ O(n) + 2T\left(\frac{n}{2}\right) & n > 1 \end{cases}$$

$$T(n) = O(n \log n)$$





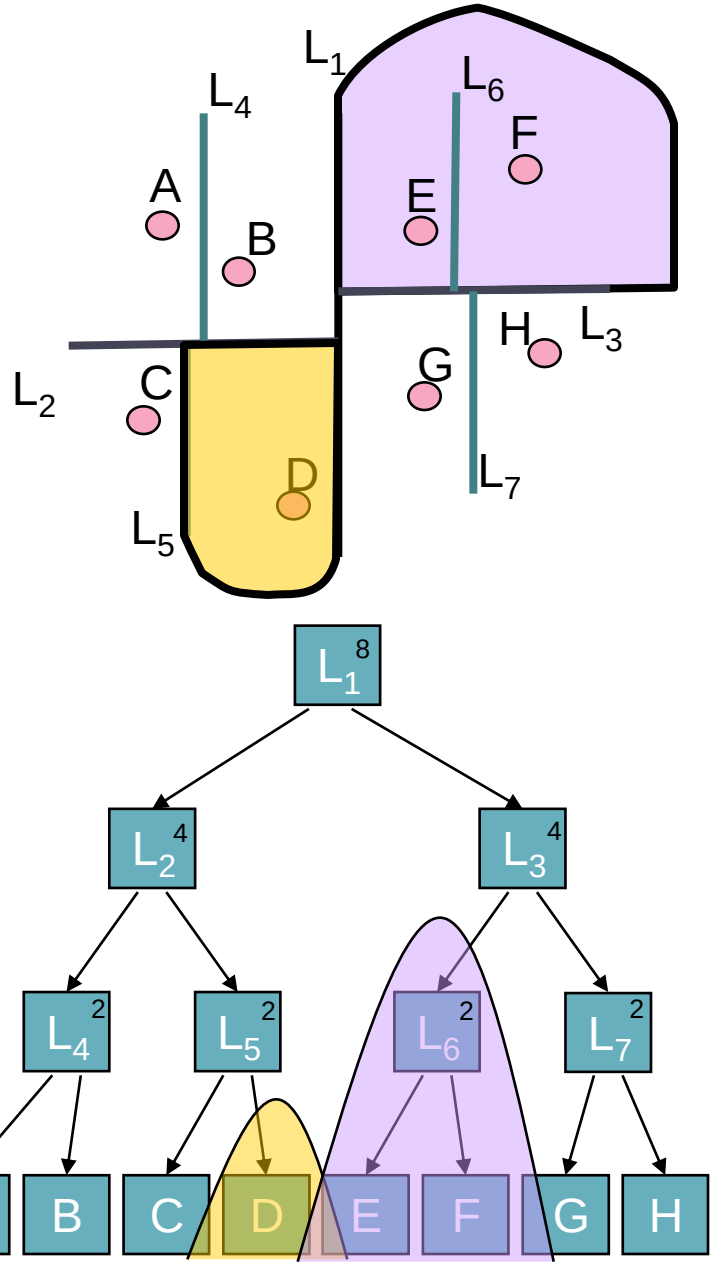
Construirea arborelui *kd* in 2D (2)

Algorithm BUILDKDTREE($P, depth$)
Input. A set of points P and the current depth *depth*.
Output. The root of a kd-tree storing P.

1. **if** P contains only one point
2. **then return** a leaf storing this point
3. **else if** *depth* is even
4. **then** Split P into two subsets with a vertical line ℓ through the median x-coordinate of the points in P. Let P_1 be the set of points to the left of ℓ or on ℓ, and let P_2 be the set of points to the right of ℓ.
5. **else** Split P into two subsets with a horizontal line ℓ through the median y-coordinate of the points in P. Let P_1 be the set of points below ℓ or on ℓ, and let P_2 be the set of points above ℓ.
6. $v_{left} \leftarrow$ BUILDKDTREE($P_1, depth + 1$)
7. $v_{right} \leftarrow$ BUILDKDTREE($P_2, depth + 1$)
8. Create a node v storing ℓ, make v_{left} the left child of v, and make v_{right} the right child of v.
9. **return** v

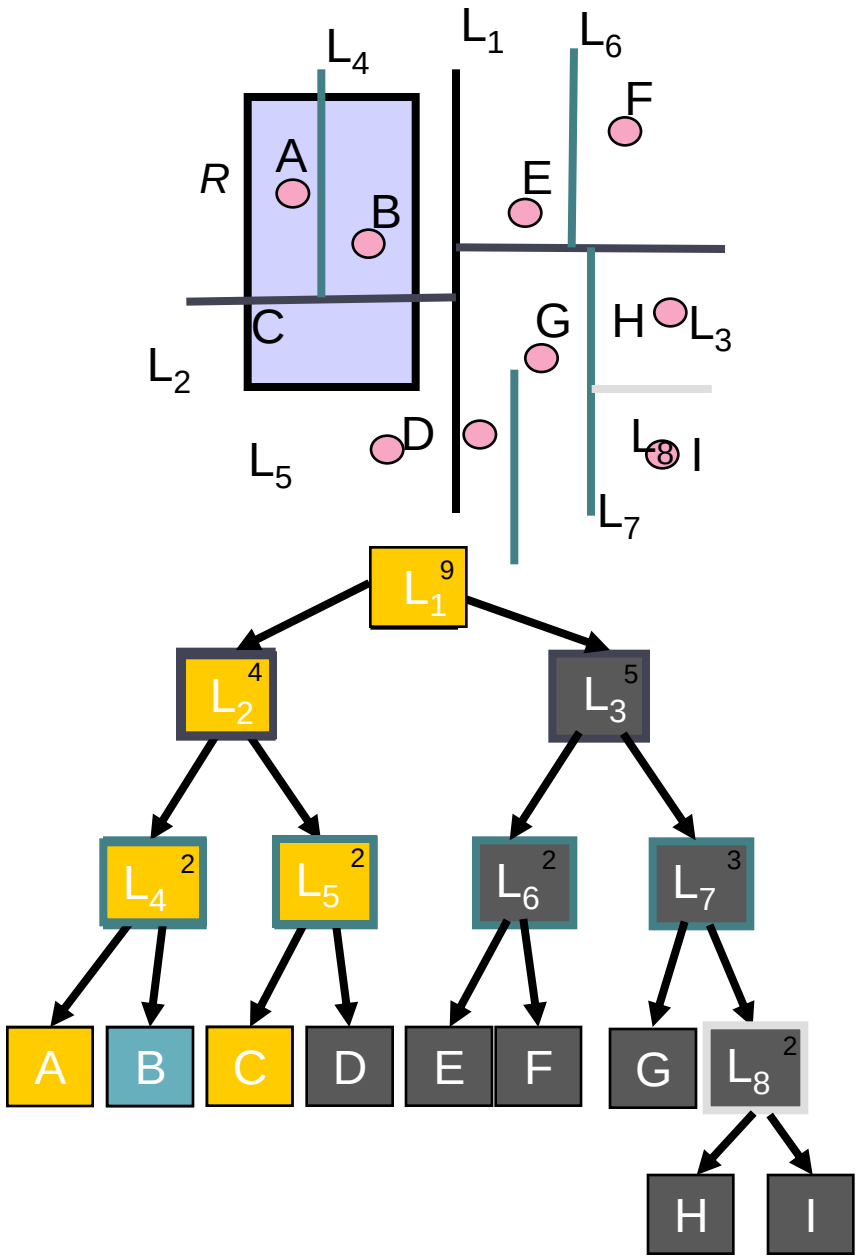
Interogare (1)

- Fiecare nod al arborelui defineste un dreptunghi paralel cu axele in plan.
- Dreptunghiul este marginit de liniile marcate de predecesorii acelui punct.
- Se eticheteaza fiecare nod cu numarul de puncte din acel dreptunghi.

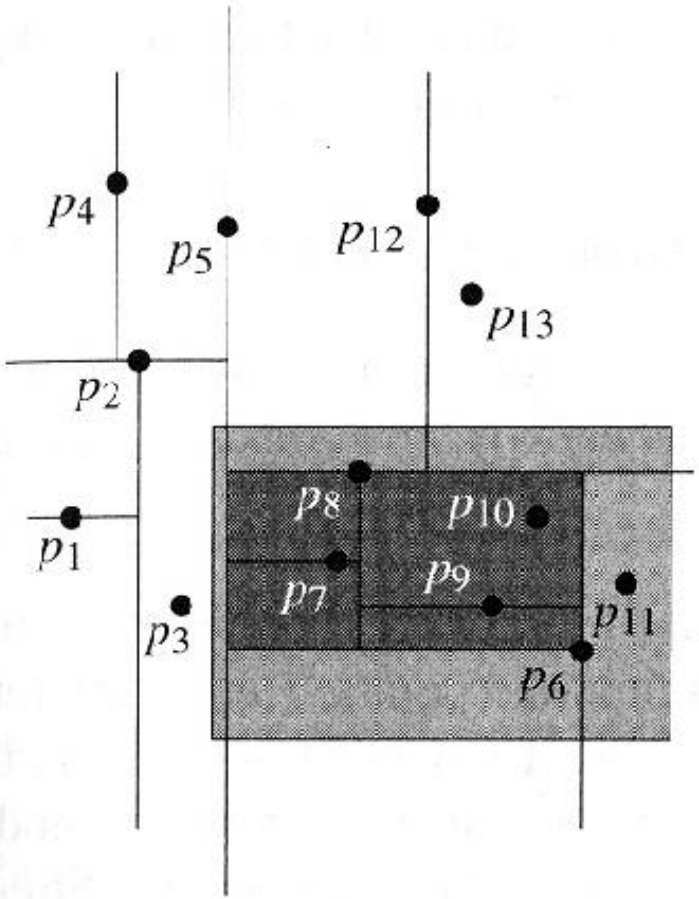


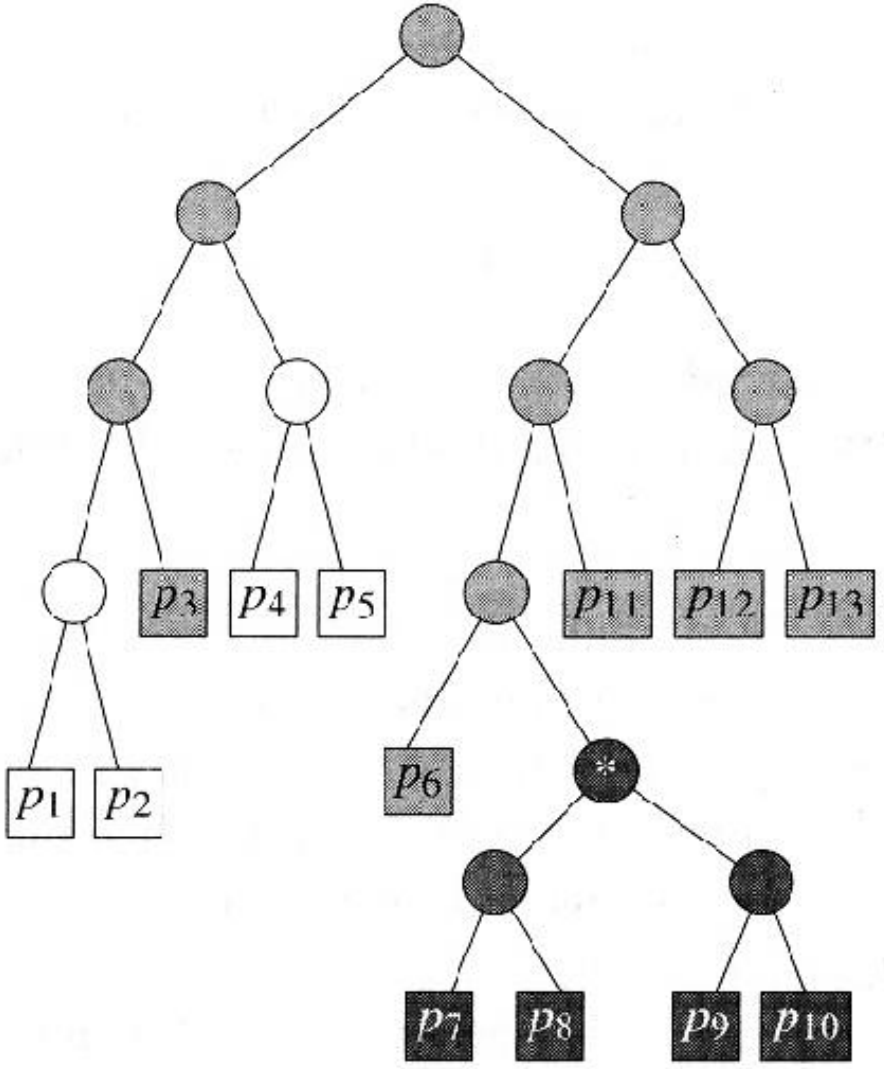
Interogare (2)

- Fiind data o interogare sub forma unui interval aliniat cu axele *R*, se cauta acest interval in arbore.
- Se traverseaza doar subarborii ce reprezinta regiuni **suprapuse** pe *R*.
- If a subtree is entirely contained in *R*:
- Daca un subarbore este continut in intregime in *R*:
  - *Numaratoare:* Se adauga valoarea sa la numaratoare
  - *Raportare:* Se raporteaza toti subarborii



Interogare: exemplu





Interogare: algoritm

Algorithm SEARCHKDTREE(ν, R)
Input. The root of a (subtree of a) kd-tree and a range R.
Output. All points at leaves below ν that lie in the range.

```
1.  if ν is a leaf
2.     then Report the point stored at ν if it lies in R.
3.     else if region(lc(ν)) is fully contained in R
4.             then REPORTSUBTREE(lc(ν))
5.             else if region(lc(ν)) intersects R
6.                     then SEARCHKDTREE(lc(ν), R)
7.          if region(rc(ν)) is fully contained in R
8.             then REPORTSUBTREE(rc(ν))
9.             else if region(rc(ν)) intersects R
10.                    then SEARCHKDTREE(rc(ν), R)
```

Interogare: complexitate

- Interogarea intoarce k noduri.
- Nodurile vizitare in algoritm sunt cele *intersectate* de dreptunghiul R dar nu continute in acesta. *Cate astfel de celule exista?*

Teorema: Fiecare latura a dreptunghiului R intersecteaza $O(\sqrt{n})$ celule ale arborelui.

Demonstratie: Se prelungeste latura pe ambele parti. In primul nivel, intersecteaza doi subarbori, iar pe nivelul urmator intersecteaza doi din cei patru subarbori, etc.

$$Q(n) = \begin{cases} 1 & n = 1 \\ 2 + 2Q\left(\frac{n}{4}\right) & n > 1 \end{cases}$$
$$= O\left(\sqrt{n}\right)$$

Timpul total de interogare: $O(\sqrt{n} + k)$.

Arbori *kd* in dimensiuni superioare

- **Pentru un spatiu d-dimensional:**
  - Acelasi algoritm.
  - Timp de constructie: $O(dn \lg n)$ deoarece timpul $O(d)$ este necesar pentru tratarea unui punct.
  - Complexitate de spatiu: $O(dn)$.
  - Complexitatea de timp a interogarii: $O(d(n^{1-1/d}+k))$.