Probleme de căutare

Mihai-Sorin Stupariu

Sem. I, 2017-2018

Motivație

Exemplu.

Baza de date a unei bănci: informații numerice referitoare la clienți: data nașterii, număr de copii, venitul lunar, valoarea depozitelor, valoarea ratelor de plată, valoarea comisioanelor plătite anual, etc. $\rightarrow$ stocarea se realizează folosind puncte dintr-un spațiu numeric d-dimensional $\mathbb{R}^d$.

Motivație

Exemplu.

Baza de date a unei bănci: informații numerice referitoare la clienți: data nașterii, număr de copii, venitul lunar, valoarea depozitelor, valoarea ratelor de plată, valoarea comisioanelor plătite anual, etc. $\rightarrow$ stocarea se realizează folosind puncte dintr-un spațiu numeric d-dimensional $\mathbb{R}^d$.

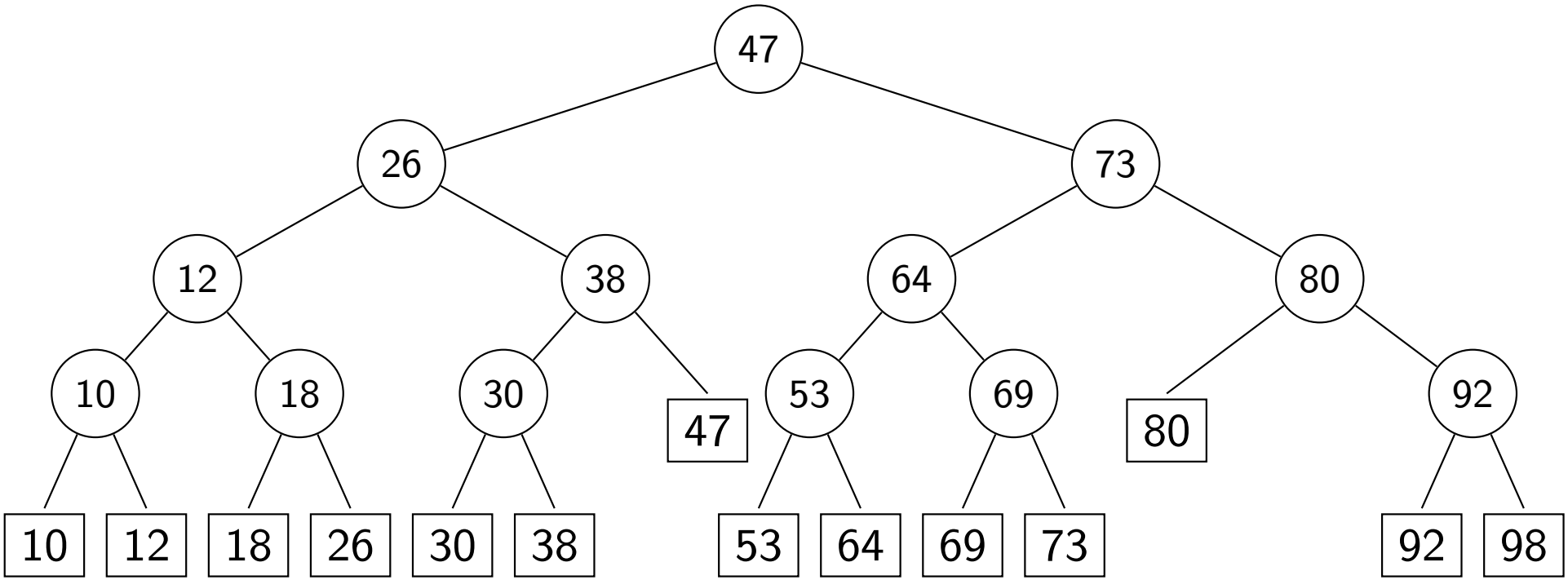
A identifica un "grup-țintă" de clienți (de exemplu pentru lansarea unui produs), având anumite caracteristici – e.g. vârsta între 30-40 ani, 2-4 copii, un venit lunar între 3000-5000 lei, etc. revine la efectuarea căutări prin care să fie determinate punctele situate într-un "paralelipiped" d-dimensional.

Căutare 1-dimensională: formularea problemei

Cadru. Fie $M = \{a_1, a_2, \ldots, a_n\}$ o mulțime de numere reale. Fie $I = [x, x'] \subset \mathbb{R}$ un interval real. Se dorește determinarea elementelor lui M situate în intervalul I.

Structura de date utilizată: Arbore binar de căutare echilibrat.

Exemplu de arbore $\mathcal{T}$



Procedura GăseşteNodSplitare $(\mathcal{T}, x, x')$

- **Input.** Un arbore binar de căutare echilibrat $\mathcal{T}$, două numere reale $x < x'$.

Procedura GăseşteNodSplitare $(\mathcal{T}, x, x')$

- **Input.** Un arbore binar de căutare echilibrat $\mathcal{T}$, două numere reale $x < x'$.
- **Output.** Nodul v în care se realizează splitarea drumurilor către x și x' sau frunza pe care ambele drumuri se încheie.

Procedura GăseşteNodSplitare $(\mathcal{T}, x, x')$

- **Input.** Un arbore binar de căutare echilibrat $\mathcal{T}$, două numere reale $x < x'$.
- **Output.** Nodul v în care se realizează splitarea drumurilor către x și x' sau frunza pe care ambele drumuri se încheie.

1. $v \leftarrow root(\mathcal{T})$

Procedura GĂSEŞTENODSPLITARE $(\mathcal{T}, x, x')$

- **Input.** Un arbore binar de căutare echilibrat $\mathcal{T}$, două numere reale $x < x'$.
- **Output.** Nodul v în care se realizează splitarea drumurilor către x și x' sau frunza pe care ambele drumuri se încheie.

1. $v \leftarrow root(\mathcal{T})$
2. **while** v nu este frunză **and** $(x' \leq x_v$ **or** $x \geq x_v)$

Procedura GĂSEŞTENODSPLITARE $(\mathcal{T}, x, x')$

- **Input.** Un arbore binar de căutare echilibrat $\mathcal{T}$, două numere reale $x < x'$.
- **Output.** Nodul v în care se realizează splitarea drumurilor către x și x' sau frunza pe care ambele drumuri se încheie.

1. $v \leftarrow root(\mathcal{T})$
2. **while** v nu este frunză **and** $(x' \leq x_v$ **or** $x \geq x_v)$
3. **do if** $x' \leq x_v$

Procedura GĂSEŞTENODSPLITARE $(\mathcal{T}, x, x')$

- **Input.** Un arbore binar de căutare echilibrat $\mathcal{T}$, două numere reale $x < x'$.
- **Output.** Nodul v în care se realizează splitarea drumurilor către x și x' sau frunza pe care ambele drumuri se încheie.

1. $v \leftarrow root(\mathcal{T})$
2. **while** v nu este frunză **and** $(x' \leq x_v$ **or** $x \geq x_v)$
3. **do if** $x' \leq x_v$
4. **then** $v \leftarrow lc(v)$

Procedura GĂSEŞTENODSPLITARE $(\mathcal{T}, x, x')$

- **Input.** Un arbore binar de căutare echilibrat $\mathcal{T}$, două numere reale $x < x'$.
- **Output.** Nodul v în care se realizează splitarea drumurilor către x și x' sau frunza pe care ambele drumuri se încheie.

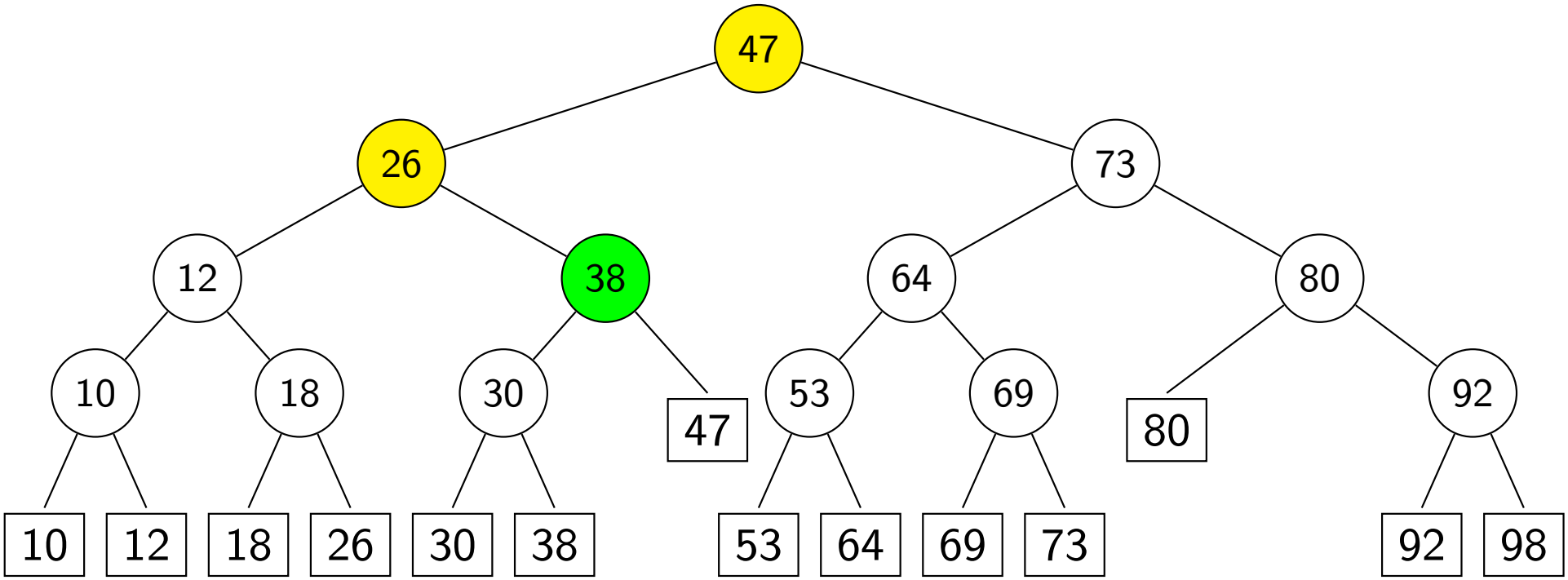
1. $v \leftarrow root(\mathcal{T})$
2. **while** v nu este frunză **and** $(x' \leq x_v$ **or** $x \geq x_v)$
3. **do if** $x' \leq x_v$
4. **then** $v \leftarrow lc(v)$
5. **else** $v \leftarrow rc(v)$

Procedura GĂSEŞTENODSPLITARE $(\mathcal{T}, x, x')$

- **Input.** Un arbore binar de căutare echilibrat $\mathcal{T}$, două numere reale $x < x'$.
- **Output.** Nodul v în care se realizează splitarea drumurilor către x și x' sau frunza pe care ambele drumuri se încheie.

1. $v \leftarrow root(\mathcal{T})$
2. **while** v nu este frunză **and** $(x' \leq x_v$ **or** $x \geq x_v)$
3. **do if** $x' \leq x_v$
4. **then** $v \leftarrow lc(v)$
5. **else** $v \leftarrow rc(v)$
6. return(v)

Exemplu de aplicare GĂSEŞTENODSPLITARE $(\mathcal{T}, 35, 40)$



În nodul $x_v = 47$ este ales $lc(v)$
În nodul $x_v = 26$ este ales $rc(v)$
În nodul $x_v = 38$ se realizează splitarea, acest nod fiind returnat

Algoritm CĂUTARE 1-DIMENSIONALĂ $(\mathcal{T}, [x, x'])$

- **Input.** Un arbore binar de căutare echilibrat $\mathcal{T}$, un interval $[x, x']$.

Algoritm CĂUTARE 1-DIMENSIONALĂ $(\mathcal{T}, [x, x'])$

- **Input.** Un arbore binar de căutare echilibrat $\mathcal{T}$, un interval $[x, x']$.
- **Output.** Toate elementele din $\mathcal{T}$ aflate în intervalul $[x, x']$.

Algoritm CĂUTARE 1-DIMENSIONALĂ $(\mathcal{T}, [x, x'])$

- **Input.** Un arbore binar de căutare echilibrat $\mathcal{T}$, un interval $[x, x']$.
- **Output.** Toate elementele din $\mathcal{T}$ aflate în intervalul $[x, x']$.

1. $v_{split} \leftarrow$ GĂSEŞTENODSPLITARE $(\mathcal{T}, x, x')$

Algoritm CĂUTARE 1-DIMENSIONALĂ $(\mathcal{T}, [x, x'])$

- **Input.** Un arbore binar de căutare echilibrat $\mathcal{T}$, un interval $[x, x']$.
- **Output.** Toate elementele din $\mathcal{T}$ aflate în intervalul $[x, x']$.

1. $v_{split} \leftarrow$ GĂSEŞTENODSPLITARE $(\mathcal{T}, x, x')$
2. **if** v_{split} este frunză
3. **then** verifică dacă elementul memorat în v_{split} trebuie raportat

Algoritm CĂUTARE 1-DIMENSIONALĂ $(\mathcal{T}, [x, x'])$

- **Input.** Un arbore binar de căutare echilibrat $\mathcal{T}$, un interval $[x, x']$.
- **Output.** Toate elementele din $\mathcal{T}$ aflate în intervalul $[x, x']$.

1. $v_{split} \leftarrow$ GĂSEŞTENODSPLITARE $(\mathcal{T}, x, x')$
2. **if** v_{split} este frunză
3. **then** verifică dacă elementul memorat în v_{split} trebuie raportat
4. **else** // Caută drumul spre x, raportează subarborii din dreapta

Algoritm CĂUTARE 1-DIMENSIONALĂ $(\mathcal{T}, [x, x'])$

- **Input.** Un arbore binar de căutare echilibrat $\mathcal{T}$, un interval $[x, x']$.
- **Output.** Toate elementele din $\mathcal{T}$ aflate în intervalul $[x, x']$.

1. $v_{split} \leftarrow$ GĂSEŞTENODSPLITARE $(\mathcal{T}, x, x')$
2. **if** v_{split} este frunză
3. **then** verifică dacă elementul memorat în v_{split} trebuie raportat
4. **else** // Caută drumul spre x, raportează subarborii din dreapta
5. $v \leftarrow lc(v_{split})$

Algoritm CĂUTARE 1-DIMENSIONALĂ $(\mathcal{T}, [x, x'])$

- **Input.** Un arbore binar de căutare echilibrat $\mathcal{T}$, un interval $[x, x']$.
- **Output.** Toate elementele din $\mathcal{T}$ aflate în intervalul $[x, x']$.

1. $v_{split} \leftarrow$ GĂSEŞTENODSPLITARE $(\mathcal{T}, x, x')$
2. **if** v_{split} este frunză
3. **then** verifică dacă elementul memorat în v_{split} trebuie raportat
4. **else** // Caută drumul spre x, raportează subarborii din dreapta
5. $v \leftarrow lc(v_{split})$
6. **while** v nu este o frunză
7. **do if** $x \leq x_v$
8. **then** RAPORTEAZĂSUBARBORE$(rc(v))$
9. $v \leftarrow lc(v)$
10. **else** $v \leftarrow rc(v)$

Algoritm CĂUTARE 1-DIMENSIONALĂ $(\mathcal{T}, [x, x'])$

- **Input.** Un arbore binar de căutare echilibrat $\mathcal{T}$, un interval $[x, x']$.
- **Output.** Toate elementele din $\mathcal{T}$ aflate în intervalul $[x, x']$.

1. $v_{split} \leftarrow$ GĂSEŞTENODSPLITARE $(\mathcal{T}, x, x')$
2. **if** v_{split} este frunză
3. **then** verifică dacă elementul memorat în v_{split} trebuie raportat
4. **else** // Caută drumul spre x, raportează subarborii din dreapta
5. $v \leftarrow lc(v_{split})$
6. **while** v nu este o frunză
7. **do if** $x \leq x_v$
8. **then** RAPORTEAZĂSUBARBORE$(rc(v))$
9. $v \leftarrow lc(v)$
10. **else** $v \leftarrow rc(v)$
11. Verifică dacă elementul din frunza v trebuie raportat

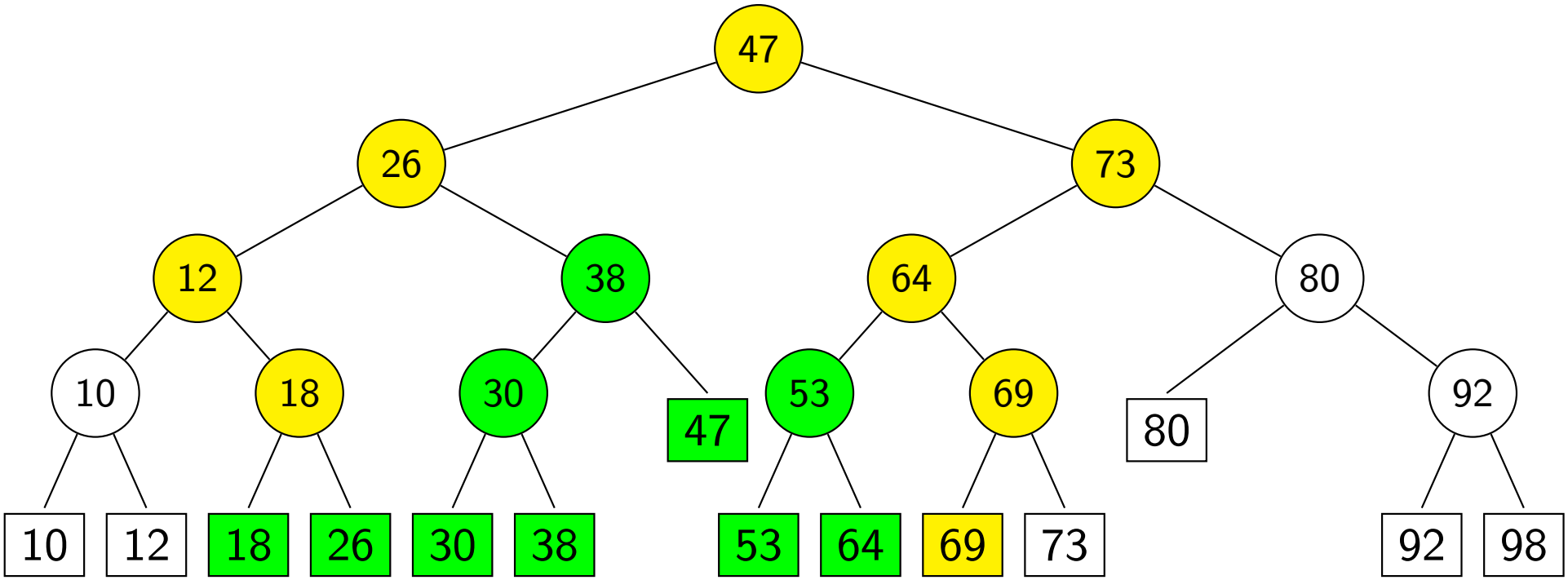
Algoritm CĂUTARE 1-DIMENSIONALĂ $(\mathcal{T}, [x, x'])$

- **Input.** Un arbore binar de căutare echilibrat $\mathcal{T}$, un interval $[x, x']$.
- **Output.** Toate elementele din $\mathcal{T}$ aflate în intervalul $[x, x']$.

1. $v_{split} \leftarrow$ GĂSEŞTENODSPLITARE $(\mathcal{T}, x, x')$
2. **if** v_{split} este frunză
3. **then** verifică dacă elementul memorat în v_{split} trebuie raportat
4. **else** // Caută drumul spre x, raportează subarborii din dreapta
5. $v \leftarrow lc(v_{split})$
6. **while** v nu este o frunză
7. **do if** $x \leq x_v$
8. **then** RAPORTEAZĂSUBARBORE$(rc(v))$
9. $v \leftarrow lc(v)$
10. **else** $v \leftarrow rc(v)$
11. Verifică dacă elementul din frunza v trebuie raportat

12.-19. Efectuează pași similari pentru x'

Aplicare CĂUTARE 1-DIMENSIONALĂ $(\mathcal{T}, [18, 68])$



Nodul de splitare este 47

Nodurile / frunzele colorate cu verde sunt raportate $\longrightarrow$ Subarborii colorați cu verde sunt raportați

Nodurile colorate cu galben sunt vizitate, fără a fi raportate

Rezultatul principal - căutare 1D

Teoremă. *Fie M o mulțime de n puncte din $\mathbb{R}$. Mulțimea M poate fi memorată într-un arbore binar de căutare echilibrat, folosind $O(n)$ memorie și cu timp de construcție $O(n \log n)$. Determinarea unor puncte dintr-un interval I poate fi realizată cu complexitate-timp $O(k + \log n)$, unde k este numărul de puncte din $M \cap I$.*

Rezultatul principal - căutare 2D

Teoremă. *Fie M o mulțime de n puncte din planul $\mathbb{R}^2$. Un arbore de intervale (range tree) pentru M necesită $O(n \log n)$ memorie și poate fi construit în timp $O(n \log n)$. Determinarea unor puncte dintr-un dreptunghi D poate fi realizată cu complexitate-timp $O(k + \log^2 n)$, unde k este numărul de puncte din $M \cap D$.*