

Structură de arbore binar

Structura în memorie:

K	Data	Pointer _{Left}	Pointer _{Right}
---	------	-------------------------	--------------------------

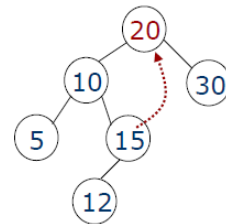
K = ID; Data = datele stocate; Pointer_{Left/Right} = pointerii spre dreapta, respective stanga

Inserare:

- detectare poziție înregistrare
- stocare înregistrare nouă într un nod liber
- legare nod la părinte

Eliminare:

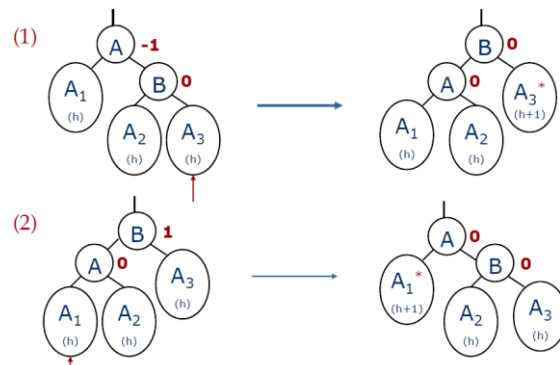
- căutare înregistrare
- 3 cazuri:
 - fără copii: pointer părinte= NULL
 - 1 copil: atașează nod copil la părinte
 - 2 copii: înlocuiește cu cel mai apropiat vecin
- adaugă nodul în lista de noduri libere

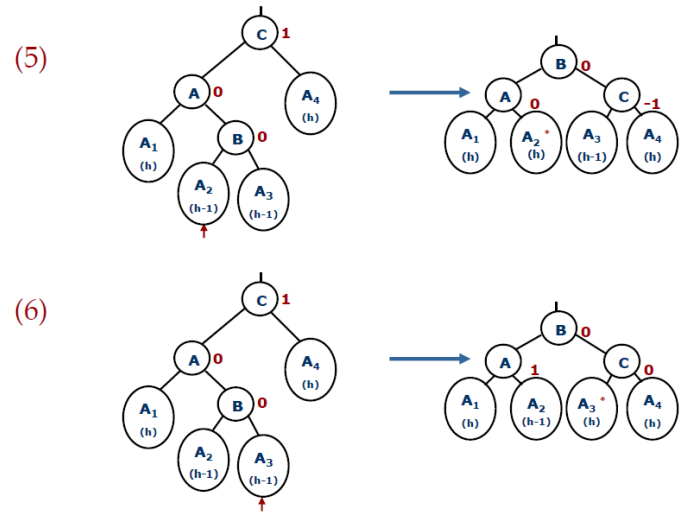
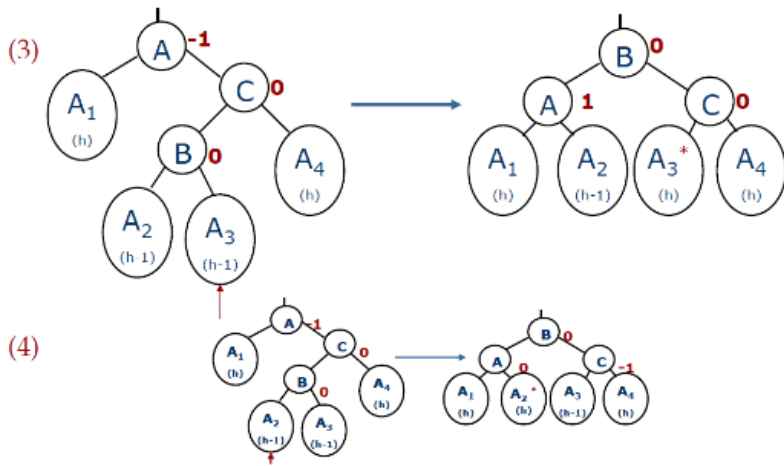


Arbori binari echilibrați

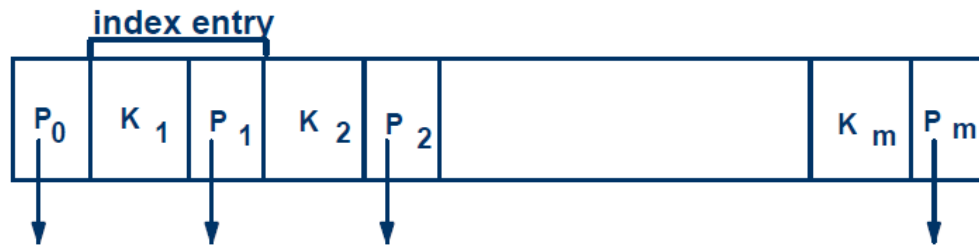
- pentru fiecare nod diferența dintre înălțimile sub arborilor săi este 0, 1 sau -1 (înălțime arbore : dimensiunea celui mai lung drum de la rădăcină la frunze)

Cazuri de dezechilibrare:

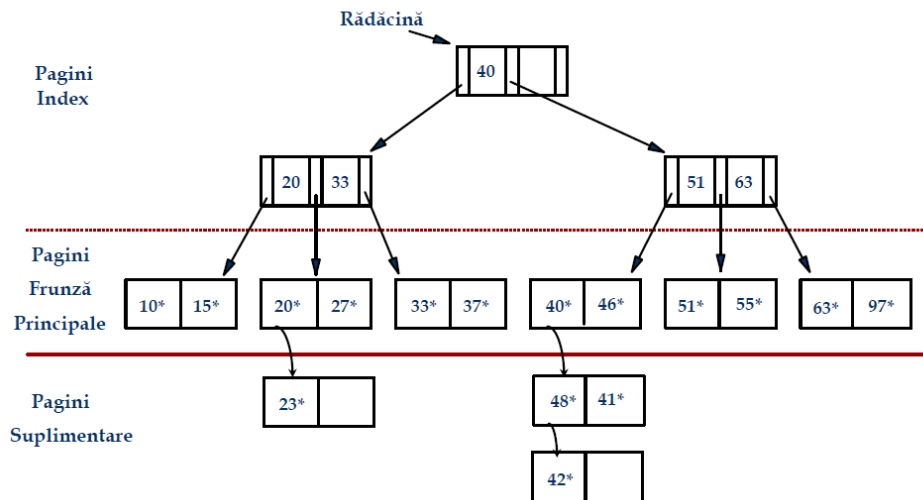




ISAM (Indexed Sequential Access Method)



Exemplu:



Organizare sub formă de Arbore-B (balanced/broad)

Proprietati:

Arbore-B de ordin m :

- Dacă nu e frunză, rădăcina va avea mereu cel puțin 2 subarbori
- Fiecare nod intern are cel puțin $\lceil m/2 \rceil$ sub-arbori (mai puțin rădăcina)
- Fiecare nod intern are cel mult m sub-arbori
- Toate frunzele sunt la același nivel
- Un nod cu p sub-arbori conține
 - $p-1$ chei ordonate (K_1, K_2, \dots, K_p)
 - T_1 conține valori $< K_1$
 - T_i conține valori între K_{i-1} și K_i
 - T_p conține valori $> K_p$



Algoritmul procedurii de inserare:

1. Localizare nod pentru inserare
2. Inserare cheie
3. Dacă nodul e plin (dimensiune depasita)
 - a. Se creaza un nou nod in care se muta cheile mai mari decat valoarea cheii mediane
 - b. Se insereaza cheia mediana in nodul parinte
 - c. Pointerul din dreapta cheii va referii noul nod, iar cel stanga refera vechiul nod ce contine valorile mai mici
4. Dacă nodul parinte e plin
 - a. Dacă nodul parinte e radacina atunci se creeaza o radacina noua
 - b. Se repeta pasul 3 pentru nodul parinte

Algoritmul de stergere:

1. Se cauta valoarea ce trebuie stearsa. Dacă se afla într-un nod intern se înlocuiește cu valoarea vecina mai mare (adică cu cea mai din stanga valoare a celei mai din stanga frunze a subarborelui drept)
2. Se repeta acest pas pana se ajunge in cazurile a. sau b.

- Dacă nodul conține valoarea de sters este rădăcina sau numărul valorilor rămase în nod este $\geq \lceil m/2 \rceil$:
 - se elimină valoarea
 - se re-aranjează valorile și pointerii din nod
 - se termină algoritmul
- Dacă numărul valorilor rămase în nod este $< \lceil m/2 \rceil$ unul dintre nodurile vecine conține $> \lceil m/2 \rceil$ valori \rightarrow redistribuire
 - se ordonează valorile din ambele noduri + valoarea separator din părinte
 - se alege valoarea mediană și se adaugă la nodul părinte, iar celelalte valori se înserează în nodul stâng, respective drept
 - algoritmul se termină
- Dacă suma valorilor din nodul din care s-a eliminat valoarea și a valorilor unuia din vecini este $< m \rightarrow$ concatenare
 - se înserează valorile ambelor noduri + valoarea separator din nodul părinte într-un singur nod
 - se repetă pasul 2. pentru nodul părinte (din care s-a eliminat valoarea separator)
 - dacă nodul părinte este rădăcina și nu mai conține valori \rightarrow nodul curent devine rădăcina

Arbori B+ : combinație între Arbori-B și ISAM

- Căutarea pornește de la rădăcină, și va fi direcționată prin comparații către o frunză
- Într-un Arbore B+ toți pointerii către înregistrări din tabele se află doar la nivelul nodurilor frunză
- Un arbore B+ poate avea mai puține nivele (sau o capacitate mai mare pentru stocarea cheilor de căutare) decât arborele B corespunzător

