Arbres

Tecnologia de la Programació

Sebastià Vila-Marta

Enginyeria de Sistemes TIC
Universitat Politècnica de Catalunya
http://itic.cat

25 de maig de 2020

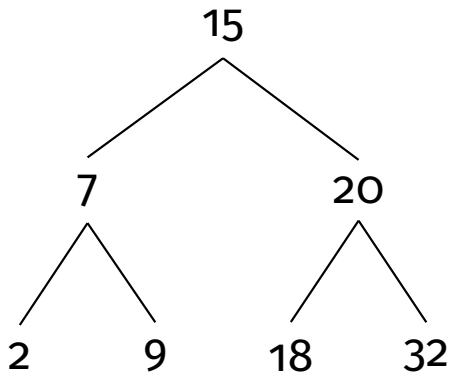
- En el tema anterior...
- Arbres binaris de cerca
- Implementació d'arbres binaris de cerca
- El cost de la cerca
- 2-3 trees

En el tema anterior...

- Hem estudiat els arbres i la seva nomenclatura.
- Hem conegut el conjunt d'operaciona habitual desl arbres i alguns exemples sobre el seu ús.
- Hem vist una implementació basada en llistes de fills per als arbres.
- Hem presentat un exemple d'ús dels arbres relacionat amb les expressions.

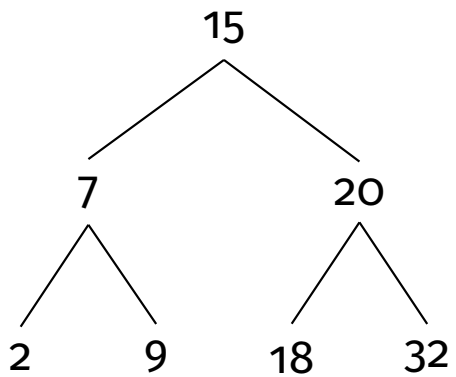
Arbre de cerca

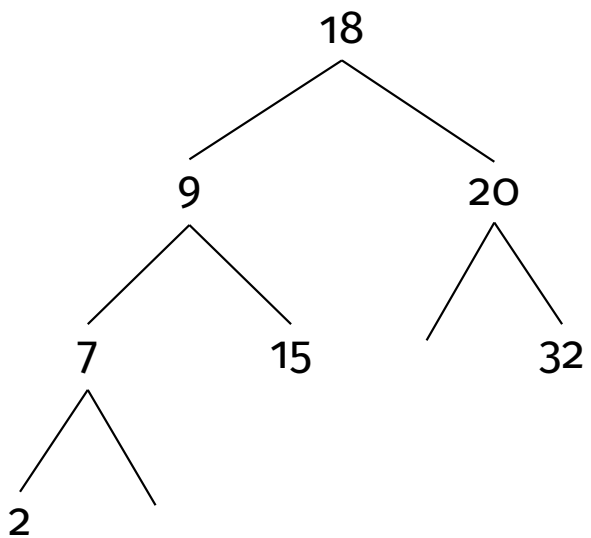
- És molt habitual usar els arbres com a estructura per emmagatzemar conjunts de dades de manera que l'operació de pertanyença sigui eficient.
- Es coneixen com arbres de cerca.
- Els més simples són els arbres binaris de cerca.
- La idea és simple: diposem el nodes de l'arbre de forma que facilitin fer cerques posteriorment.
- Imposem que:
  - Els elements menors que l'arrel siguin al subarbre esquerre.
  - El elements més grans que



Definició d'arbre binari de cerca

- Sigui $C = \{k_0, k_1, \ldots, k_n\}$ un conjunt de valors k_i tals que $\forall i, j : i \neq j : k_i \neq k_j$ i en el que està ben definida una relació d'ordre total. Els valors de C els anomenarem claus.
- Un arbre de cerca binari per al conjunt C és un arbre $\mathcal{A}$ en que els nodes tenen associat un valor de C i:
  - $e \in \mathcal{A} \iff e \in C$
  - $\forall s : s \in \mathcal{A} : (\forall v : v \in \mathit{left}(s) : v < s)$
  - $\forall s : s \in \mathcal{A} : (\forall v : v \in \mathit{right}(s) : v > s)$





Operacions d'un arbre binari de cerca I

- Les operacions fonamentals sobre un arbre de cerca són les següents:
  - insereix(self,k) Insereix en l'arbre self el valor k.
  - cerca(self,k) Retorna **True** si l'arbre conté k.
  - esborra(self,k) Esborra k de l'arbre self. Si self no contenia k, no fa res.

```
class ArbreCerca(object):
    def __init__(self):
        self.v = None
        self.d = None
        self.e = None
```

Operació de cerca I

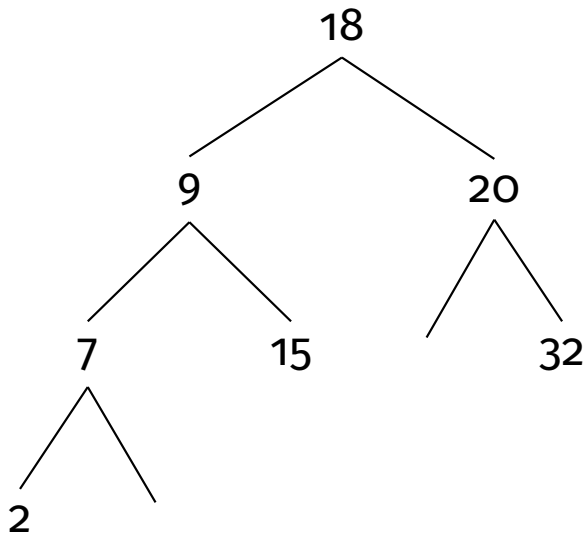
```
def cerca(self, k):
    if self.v == None:
        return False
    else:
        if self.v == k:
            return True
        elif self.v > k:
            return self.d.cerca(k)
        else:
            return self.e.cerca(k)
```

Operació d'inserció I

```
def insereix(self, k):
    if self.v == None:
        self.v = k
    else:
        if self.v == k:
            raise Exception('Repetit')
        elif self.v > k:
            return self.d.insereix(k)
        else:
            return self.e.insereix(k)
```

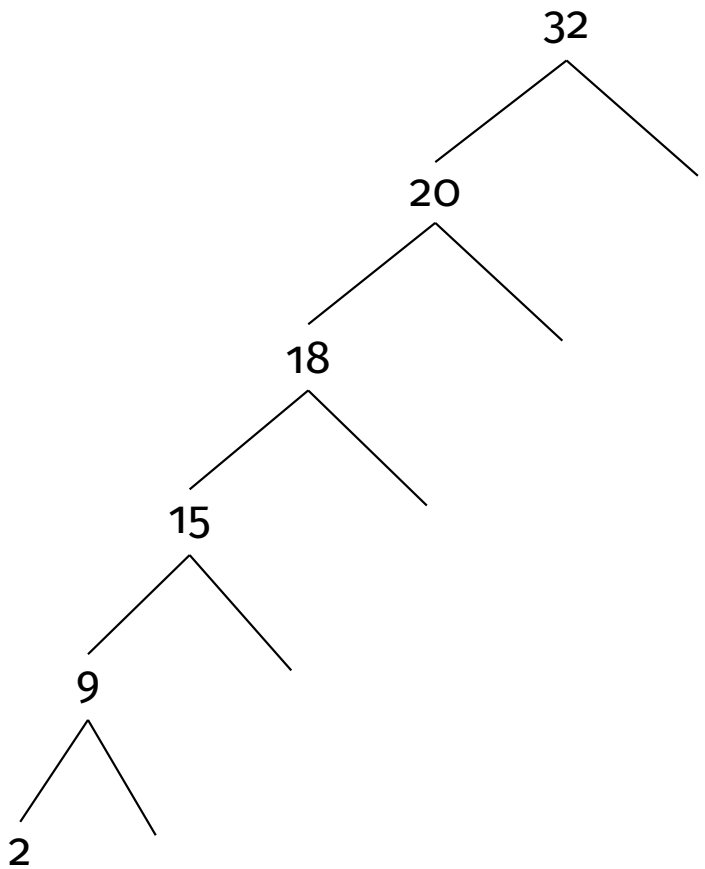
Cost de la cerca

- Sigui $C = \{k_0, k_1, \ldots, k_n\}$ un conjunt de claus i $\mathcal{A}$ un arbre de cerca de C.
- La profunditat mínima d'$\mathcal{A}$ és $\log_2 \|C\|$, però pot ser superior en alguns casos.
- Com la cerca, en al cas pitjor, fa tants passos com el camí més llarg des de l'arrel, el cost assimptòtic en cas pitjor de la cerca és de $O(\log_2 n)$ sent n el nombre d'elements del conjunt C.



Cost de la cerca (II)

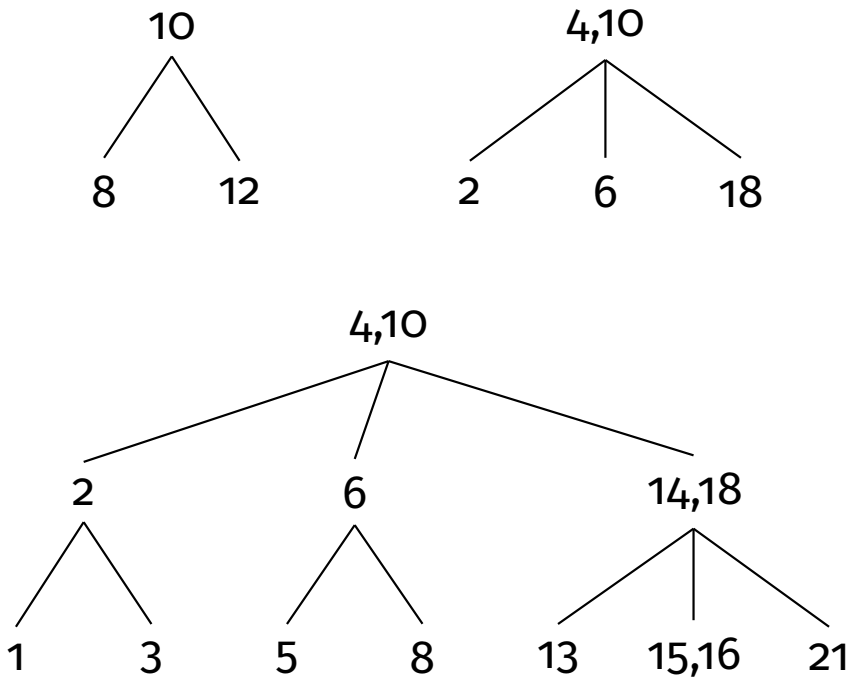
- Els arbre de cerca poden degenerar en (quasi) llistes si l'ordre d'inserció és desfavorable.
- Exemple: 32, 20, 18, 15, 9, 2.
- Aleshores el cost de cerca esdevé $O(n)$.
- Per que l'arbre de cerca sigui de cost logaritmica cal evitar que degeneri.
- Diverses estratègies permeten mantenir arbres equilibrats:
  - 2-3 trees
  - AVL trees (Adelson-Velsky and Landis)



2-3 trees

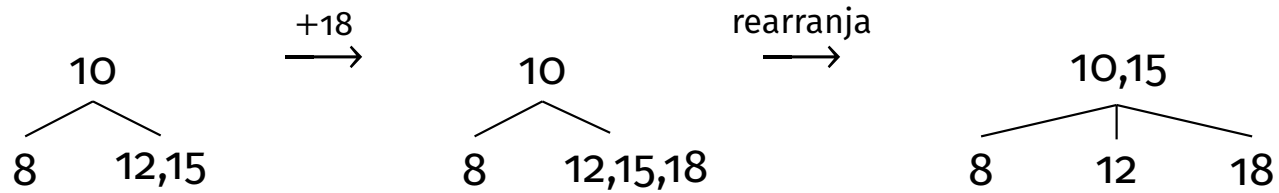
- Són una aportació de John Hopcroft. Es mantenen equilibrats durant les insercions.
- Un 2-3 tree és un arbre ternari. Hi ha dues menes de nodes interns:
  - 2-nodes: tenen dos fills.
  - 3-nodes: tenen tres fills.
- Els fills de tot node tenen la mateixa alçada sempre.
- Un arbre $\mathcal{A}$ és un 2-3 tree ssi:
  1. $\mathcal{A}$ és un arbre buit.
  2. $\mathcal{A}$ és un 2-node amb valor a i el seus fills p i q, cas que en tingui, cumpleixen:
     1. p i q tenen la mateixa alçada
     2. $E(p) < a \leq E(q)$ on $E(x)$ denota els elements de l'arbre x.
  3. $\mathcal{A}$ és un 3-node amb valors a i b i el seus fills p, q i r, cas que en tingui, cumpleixen:
     1. p, q i r tenen la mateixa alçada
     2. $E(p) < a \leq E(q) < b \leq E(r)$

2-3 trees (II)

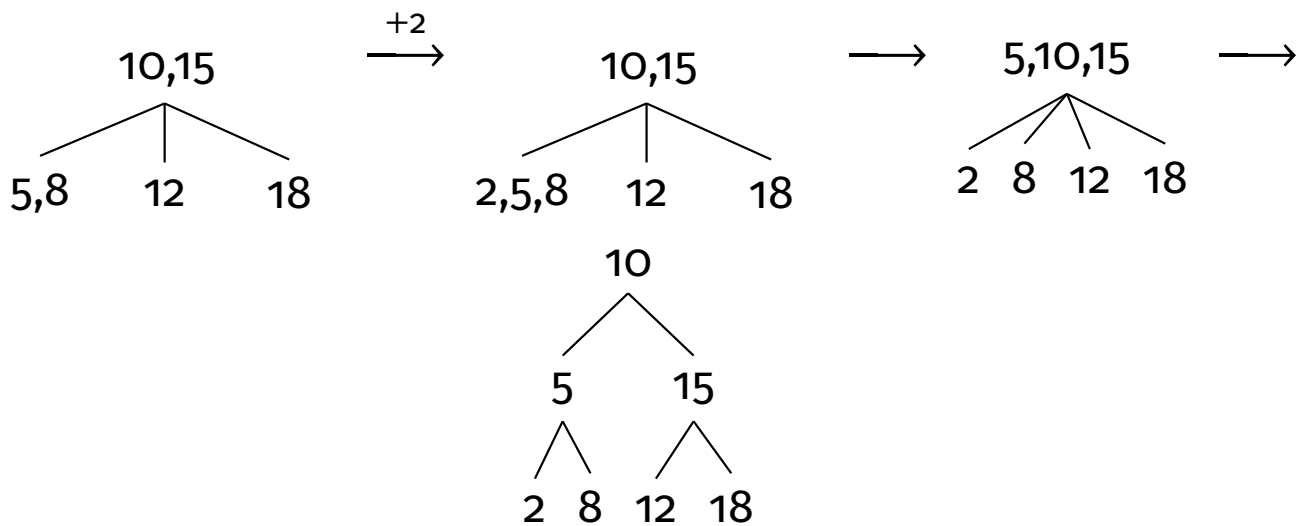


2-3 tree (III)

- La inserció aprofita la diversitat de nodes per mantenir equilibri.
- Inserció en un 2-node només afegeix valor a un node.
- Inserció en un 3-node amb pare 2-node:



- Inserció en un 3-node amb pare 3-node:



I recursivament cap amunt fins l'arrel.