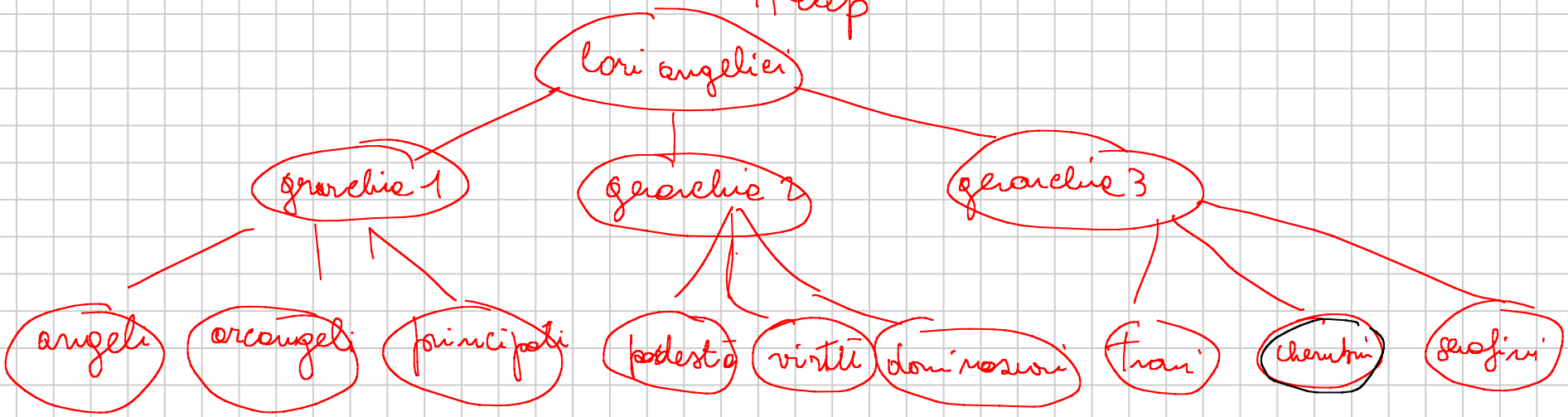
- Alberi di decisione — per il calcolo dei limiti inf.
- Alberi di ricorsione — per lo studio di eq. di ricorrenza e per la simulaz. di algoritmi Divide et Impera
- Albero come struttura dati (per rendere più efficienti operazioni)

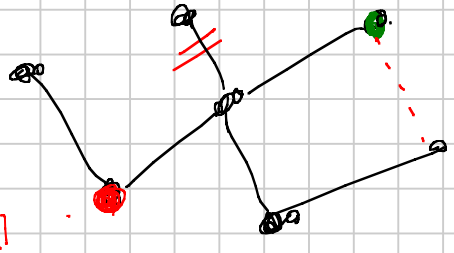
Heap



Alberi per rappresentare informazioni di tipo gerarchico

Alberi come reti di connessione

albero libero



Albero è la connessione globale col numero minimo di connessioni tra i nodi

$|E| = |V| - 1$ proprietà vera per ogni albero

Shortest Paths Tree

Routing di messaggi

Alberi in natura

Albero: è un Grafo connesso privo di cicli

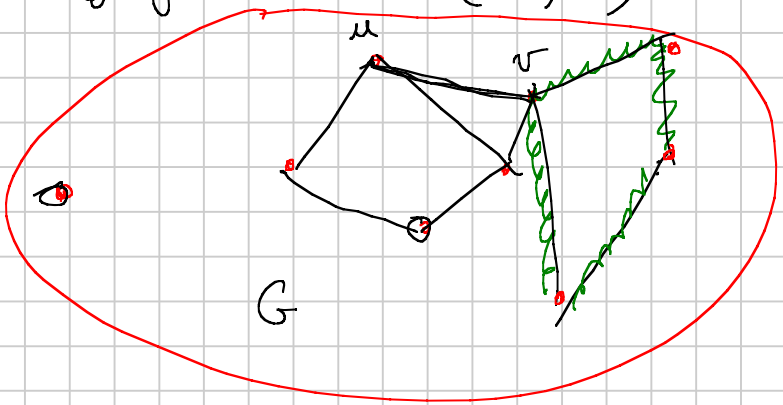
Grafo $G = (V, E)$

Vertices (nodes) Edges (Arcs)

Vertici, nodi Spigoli, archi

$|V| = n$

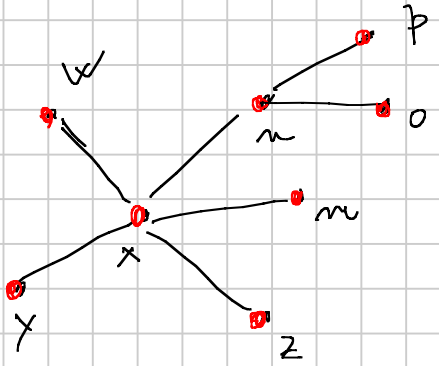
$|E| = m$ di archi (u, v)



G è connesso se tutti i suoi vertici sono raggiungibili

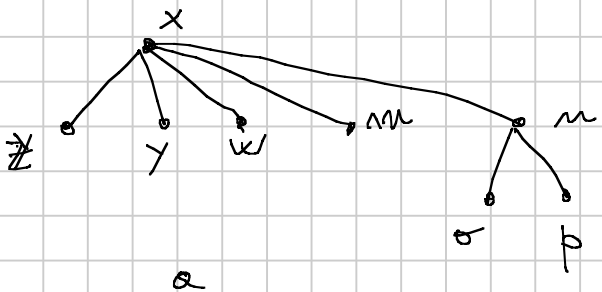
un ciclo un percorso formato da archi che inizia e termina in un vertice.

albero libero



$|E| = |V| - 1$

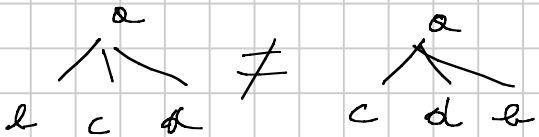
selezione di x come radice



i nodi nel livello possono avere un ordinamento o meno

⇓

alberi ordinati

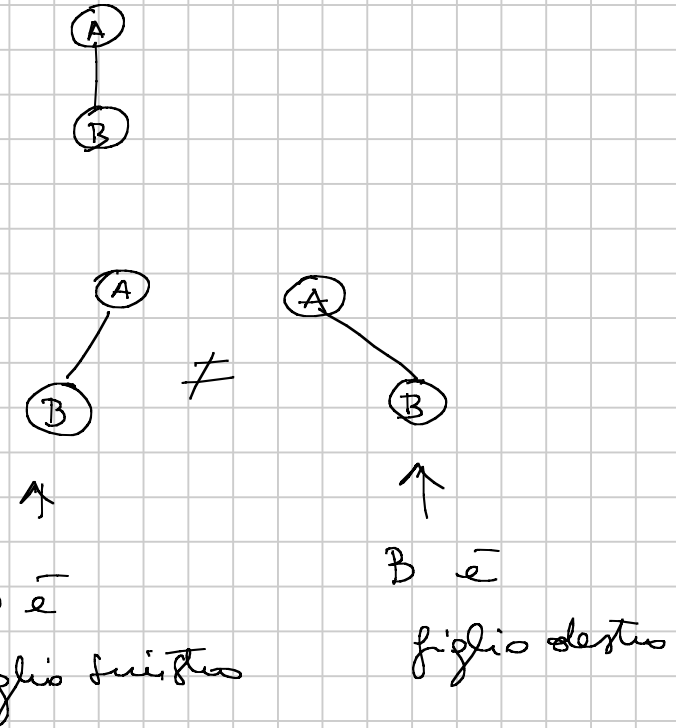


alberi binari (alberi n-ari)

Def. ricorsiva

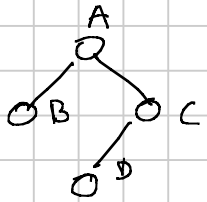
- vuoto
- i suoi nodi si possono dividere in 3 sottoinsiemi
  - radice
  - sottoalbero sinistro
  - sottoalbero destro

} alberi binari



Rappresentazione in memoria di alberi binari con n nodi.

1) Rappresentazione in array di n elementi



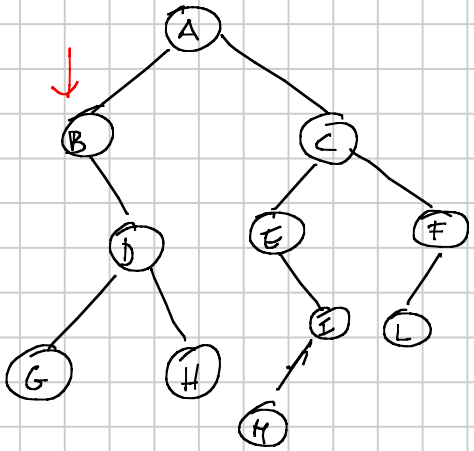
A	B	C	∅	∅	D
1	2	3	4	5	6

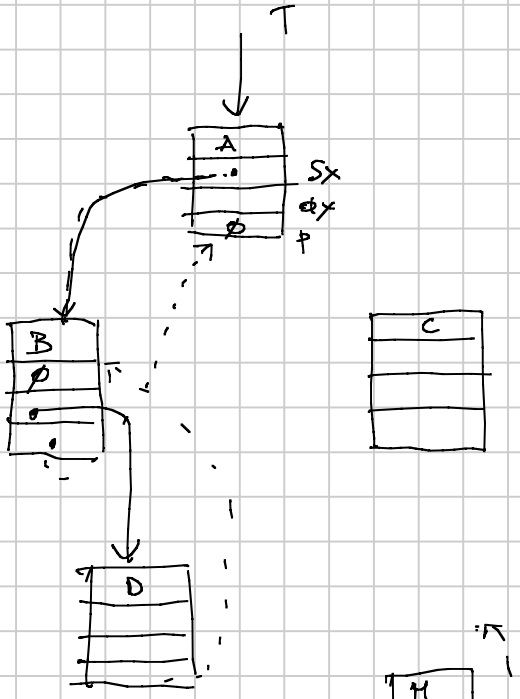
uso inefficiente della memoria.

caso pessimo: spreco esponenziale in n

non si usa a meno di avere informazioni molto precise sull'albero (come Heap)

2) Con puntatori





nodo Key
punt_sin sx
punt_des dx
punt padre p

T.Key = A

T = T.sx

Operazioni tipiche su alberi binari

1) Dimensione dell'albero (numero dei nodi)

L'albero è conosciuto all'esterno solo col puntatore alla radice.

2) Profondità di un nodo v = distanza del nodo dalla radice

3) Altezza di un albero (o di un nodo) = distanza della radice in numero di archi dalla foglia più lontana

Divide et Impera sugli alberi binari

```
Dimensione (u):
if (u == NULL)  return 0;
else {
    DimSx = Dimensione (u.sx);
    Dimdx = Dimensione (u.dx);
    dim = DimSx + Dimdx + 1;
    return dim;
}
```

Terminazione

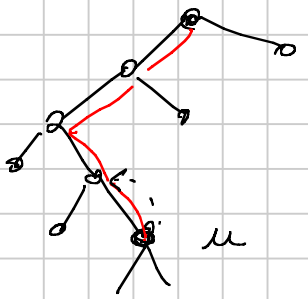
ricorsione

combinazione

Determinare la profondità di un nodo di puntatore u

```
Profondità (u):
p = 0;
while (u.px != NULL) {
    p = p + 1;
    u = u.px
}
```

finché non raggiungo la radice



```
Altezza (u)
if (u == NVLL)
        return -1;
else {
  - altsx = Altezza(u.sx);
  - altdx = Altezza(u.dx);
  - alt = max(altsx, altdx) + 1;
    return alt;
}
```

terminazione

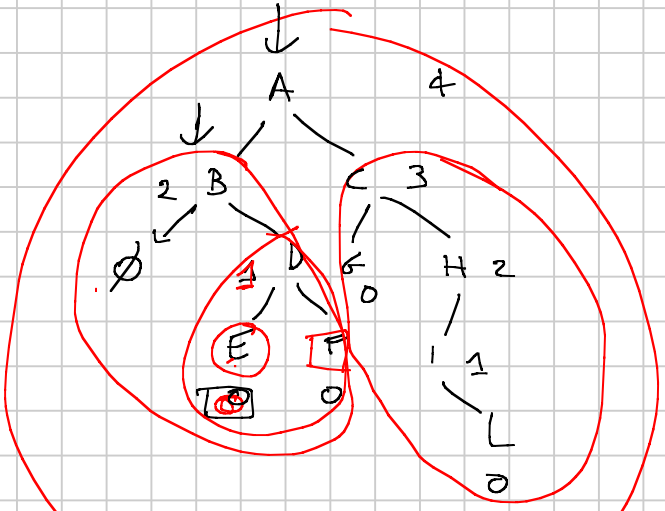
ricorsione

ricombinazione

E
∅
∅

$T(n) = 2T\left(\frac{n}{2}\right) + \Theta(1) = \Theta(n)$

Calcola l'altezza della radice dell'albero di puntatore u con metodo divide et impera



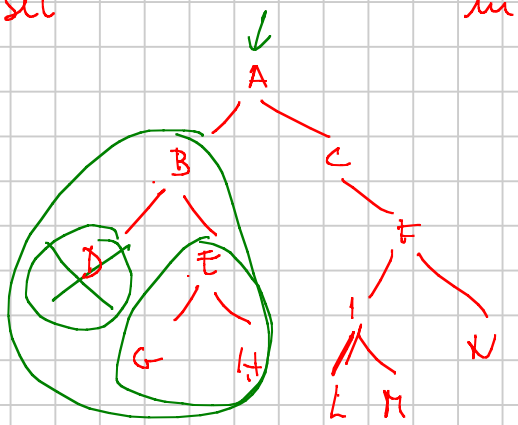
$T(n) = T(n_s) + T(n_d) + \Theta(1)$

Titolo nota 15/04/2016

Recuperare tutta l'informazione associata all'albero

Visite — Visit

- in ordine anticipato — pre-visit
- in ordine simmetrico — in-visit
- in ordine posticipato o differito — post-visit



A B D E G H C F I L M N

Visita in ordine anticipato

se l'albero non è vuoto

1) Esamino la radice
2) Visita il sott. sin in ordine anticipato
3) Visita il sott. destro

D B G E H A C L I M F N

D G H E B L M I N F C A

```
IN-Visit (u)
if (u != NULL) {
    IN-Visit (u.sx);
    Print (u.key);
    IN-Visit (u.dx);
}
```

Visita in ordine simmetrico

```
POST-VISIT(u);
if (u != NULL) {
    POST-Visit (u.sx);
    POST-Visit (u.dx);
    Print (u.key);
}
```

Visita in ordine differito

$$T(n) = T(n_s) + T(n_d) + 1$$

n_s e n_d = numero di nodi del sottoalbero sinistro e del sott. destro

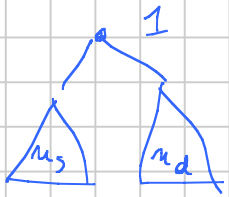
$T(n) = n$

Prova per induzione

base: $n = 1$ $\quad T(1) = 1$ $\quad$ vero

ipotesi induttiva: posto vero per tutti gli $n' < n$

$$T(n) = n_s + n_d + 1 = n$$



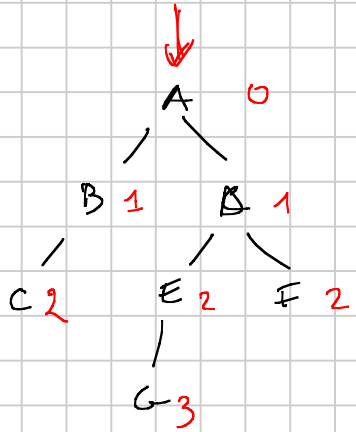
$$n = n_s + n_d + 1$$

Algoritmi su alberi "tipo" Divide et Impera

richiedono tempo lineare

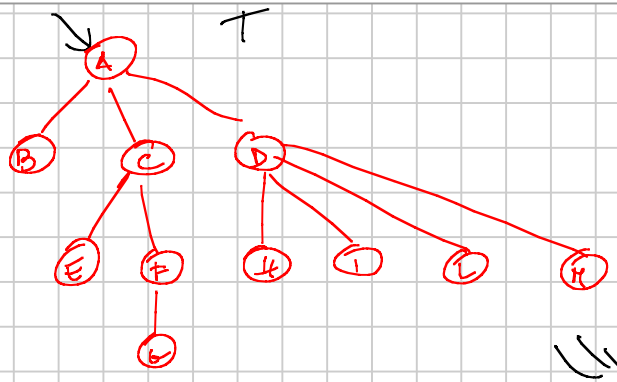
Esercizio: calcolare il numero di foglie di un albero binario di puntatori.

Esercizio Calcolo la profondità di tutti i nodi dell'albero di partenza r.

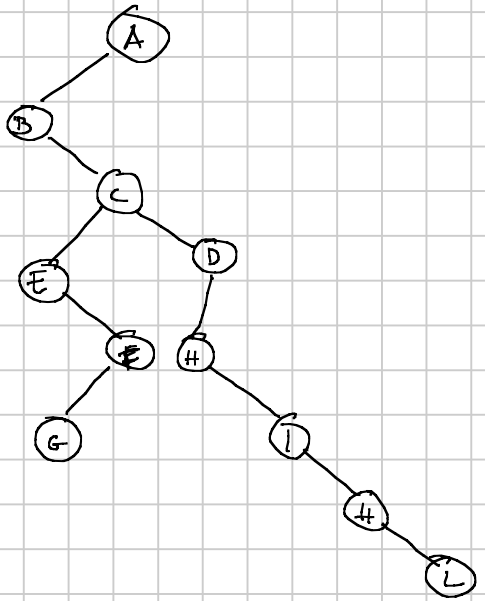


Titolo nota 15/04/2016

Rappresentazione alberi



non sappiamo il n° max di figli



Regola di trasformazione da albero ri albero binario

Le radici corrispondono

$\forall$ nodo

il figlio sinistro è il primo figlio

il figlio destro è il successivo fratello