Alberi n-ari: specifiche sintattiche e semantiche. Realizzazioni. Visita di alberi n-ari.

Algoritmi e Strutture Dati + Lab
A.A. 14/15

Informatica
Università degli Studi di Bari "Aldo Moro"

Nicola Di Mauro

ALBERO N-ARIO

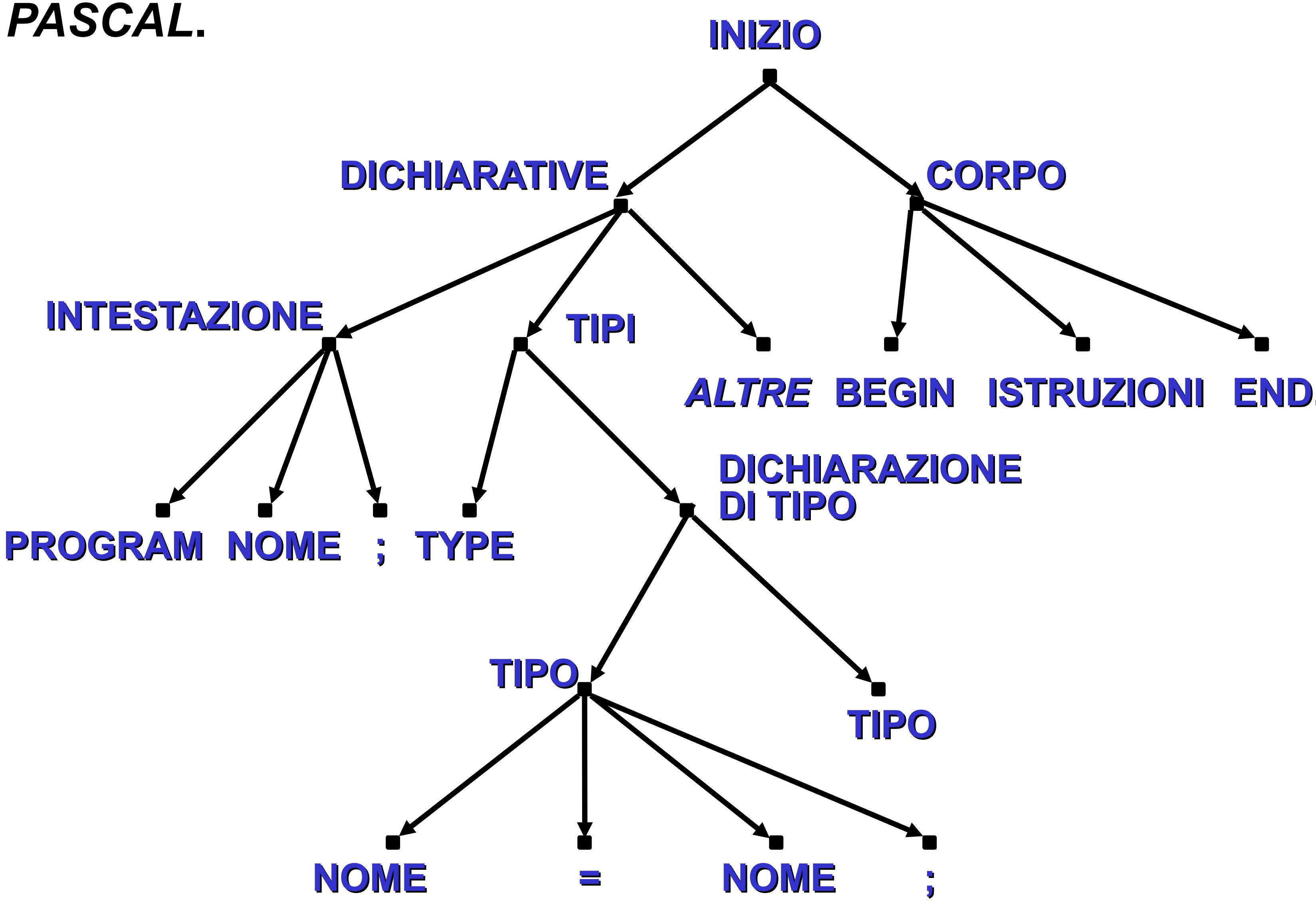
DEFINIZIONE:

UN ALBERO È UN *GRAFO ORIENTATO* CHE O È VUOTO OPPURE HA LE SEGUENTI CARATTERISTICHE:

- **ESISTE UN NODO R, DETTO RADICE, SENZA PREDECESSORI, CON n ($n \geq 0$) NODI SUCCESSORI a_1, a_2, …, a_n;**
- **TUTTI GLI ALTRI NODI SONO RIPARTITI IN n SOTTOALBERI MUTUAMENTE DISGIUNTI T_1, T_2, …, T_n AVENTI RISPETTIVAMENTE a_1, a_2, …, a_n COME RADICE.**

L'ALBERO N-ARIO È UN TIPO ASTRATTO DI DATI UTILIZZATO PER RAPPRESENTARE RELAZIONI GERARCHICHE TRA OGGETTI. NELLA DEFINIZIONE DATA ABBIAMO ASSUNTO CHE SUI FIGLI DI OGNI NODO SIA DEFINITA UNA RELAZIONE D'ORDINE (*ALBERI ORDINATI*).

<u>*ESEMPIO*</u>: APPLICAZIONE DEGLI ALBERI PER DEFINIRE LA GRAMMMATICA DEL LINGUAGGIO *PASCAL*.



SPECIFICA SINTATTICA

Tipi: ***albero, boolean, nodo***

Operatori:

CREAALBERO: ( ) $\rightarrow$ ***albero***
ALBEROVUOTO: (***albero***) $\rightarrow$ ***boolean***
INSRADICE: (***nodo, albero***) $\rightarrow$ ***albero***
RADICE: (***albero***) $\rightarrow$ ***nodo***
PADRE: (***nodo, albero***) $\rightarrow$ ***nodo***
FOGLIA: (***nodo, albero***) $\rightarrow$ ***boolean***
PRIMOFIGLIO: (***nodo, albero***) $\rightarrow$ ***nodo***
ULTIMOFRATELLO: (***nodo, albero***) $\rightarrow$ ***boolean***
SUCCFRATELLO: (***nodo, albero***) $\rightarrow$ ***nodo***
INSPRIMOSOTTOALBERO:(***nodo,albero,albero***)$\rightarrow$***albero***
INSSOTTOALBERO: (***nodo, albero, albero***) $\rightarrow$ ***albero***
CANCSOTTOALBERO: (***nodo, albero***) $\rightarrow$ ***albero***

SPECIFICA SEMANTICA

Tipi:

albero=insieme degli alberi ordinati T=<N,A> in cui ad ogni nodo n in N è associato il livello(n);
boolean=insieme dei valori di verità;
nodo=insieme qualsiasi (non infinito).

Operatori:

CREAALBERO=T'
POST: T' = (Ø,Ø) = Λ (ALBERO VUOTO)
ALBEROVUOTO(T)=b
POST: b=VERO SE T=Λ
b=FALSO, ALTRIMENTI
INSRADICE(u, T) =T'
PRE: T=Λ
POST: T'=(N,A), N={u}, LIVELLO(u)=0, A=Ø
RADICE(T) =u
PRE: T$\neq\Lambda$
POST: u$\Rightarrow$RADICE DI T$\Rightarrow$LIVELLO(u)=0
PADRE(u, T) =v
PRE: T$\neq\Lambda$, u$\in$ N, LIVELLO(u)>0
POST: v È PADRE DI u, <v,u>$\in$ A
LIVELLO(u)=LIVELLO(v)+1

FOGLIA(u, T) =b

PRE: $T \neq \Lambda$, $u \in N$

POST: b=VERO SE $\neg\exists$ $v \in N$ ∍' $<u,v> \in A$ $\wedge$

$\wedge$ LIVELLO(v)=LIVELLO(u)+1

b=FALSO, ALTRIMENTI

PRIMOFIGLIO(u, T) =v

PRE: $T \neq \Lambda$, $u \in N$, FOGLIA(u, T)=FALSO

POST: $<u,v> \in A$, LIVELLO(v)=LIVELLO(u)+1

v È PRIMO SECONDO LA RELAZIONE

D'ORDINE STABILITA TRA I FIGLI DI u

ULTIMOFRATELLO(u, T) =b

PRE: $T \neq \Lambda$, $u \in N$

POST: b=VERO SE NON ESISTONO ALTRI FRATELLI DI u CHE LO SEGUONO NELLA RELAZIONE D'ORDINE

b=FALSO, ALTRIMENTI

SUCCFRATELLO(u, T) =v

PRE: $T \neq \Lambda$, $u \in N$, ULTIMOFRATELLO(u, T)=FALSO

POST: v È IL FRATELLO DI u CHE LO SEGUE NELLA RELAZIONE D'ORDINE

INSPRIMOSOTTOALBERO(u, T, T') =T''

PRE: $T \neq \Lambda$, $T' \neq \Lambda$, $u \in N$

POST: T'' È OTTENUTO DA T AGGIUNGENDO L'ALBERO T' LA CUI RADICE r' È IL NUOVO PRIMOFIGLIO DI u

INSSOTTOALBERO(u, T, T') =T''

PRE: $T \neq \Lambda$, $T' \neq \Lambda$, $u \in N$, u NON È RADICE DI T

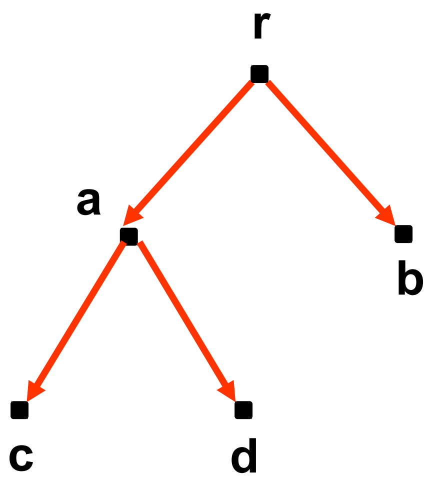
POST: T'' È L'ALBERO OTTENUTO DA T AGGIUNGENDO IL SOTTOALBERO T' DI RADICE r' (CIOÈ r' DIVENTA IL NUOVO FRATELLO CHE SEGUE u NELLA RELAZIONE D'ORDINE)

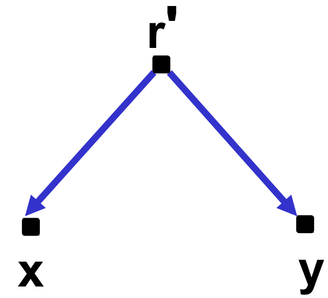
CANCSOTTOALBERO(u, T) =T'

PRE: $T \neq \Lambda$, $u \in N$

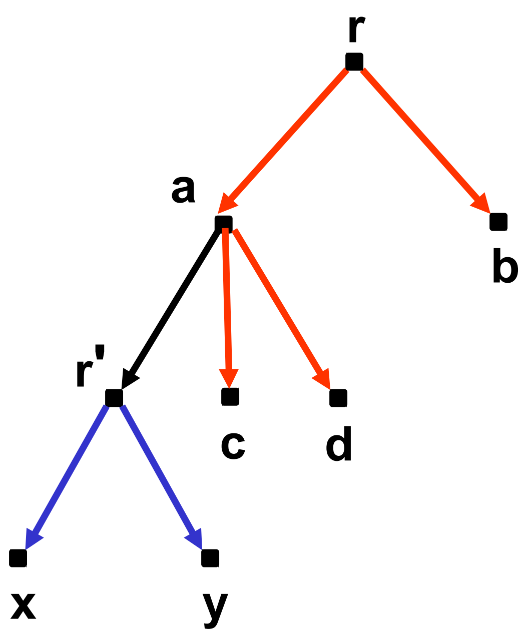
POST: T' È OTTENUTO DA T TOGLIENDOVI IL SOTTOALBERO DI RADICE u (CIOÈ u E TUTTI I SUOI DISCENDENTI)

ESEMPI DI INSERIMENTI

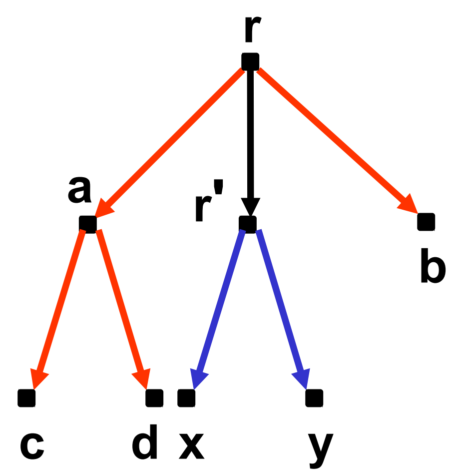




INSPRIMOSOTTOALBERO(a,T,T')

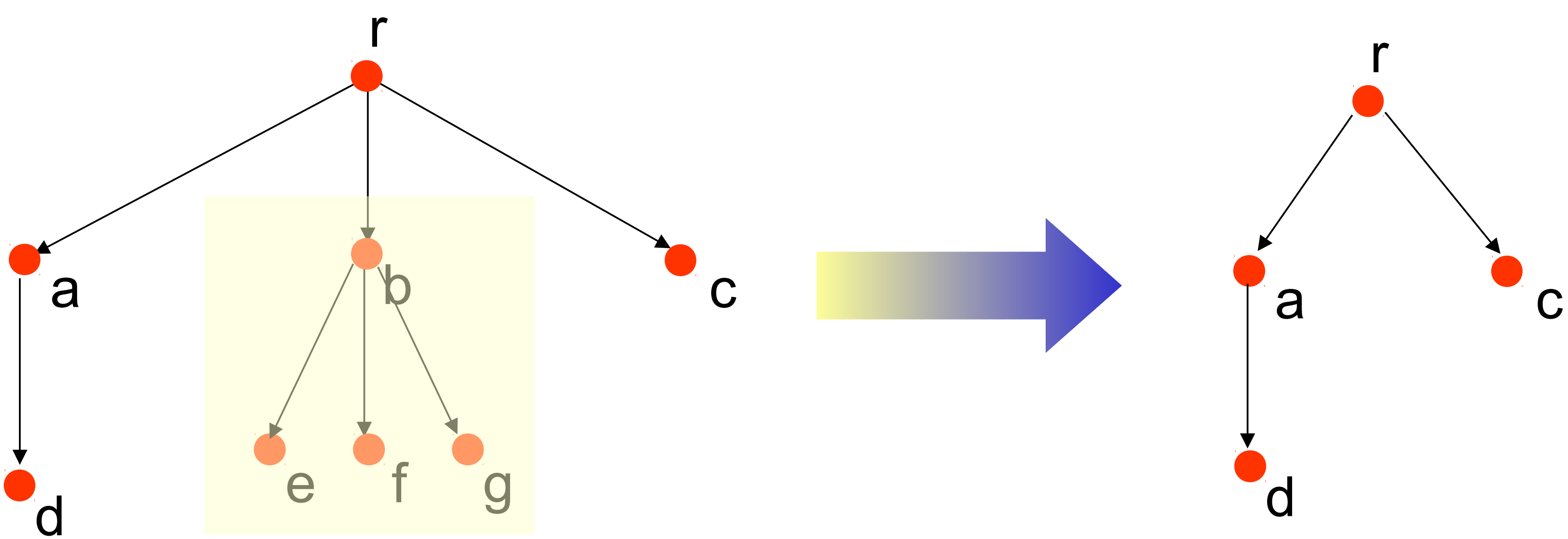


INSSOTTOALBERO(a,T,T')



ESEMPIO DI CANCELLAZIONE

CANCSOTTOALBERO(b,T)



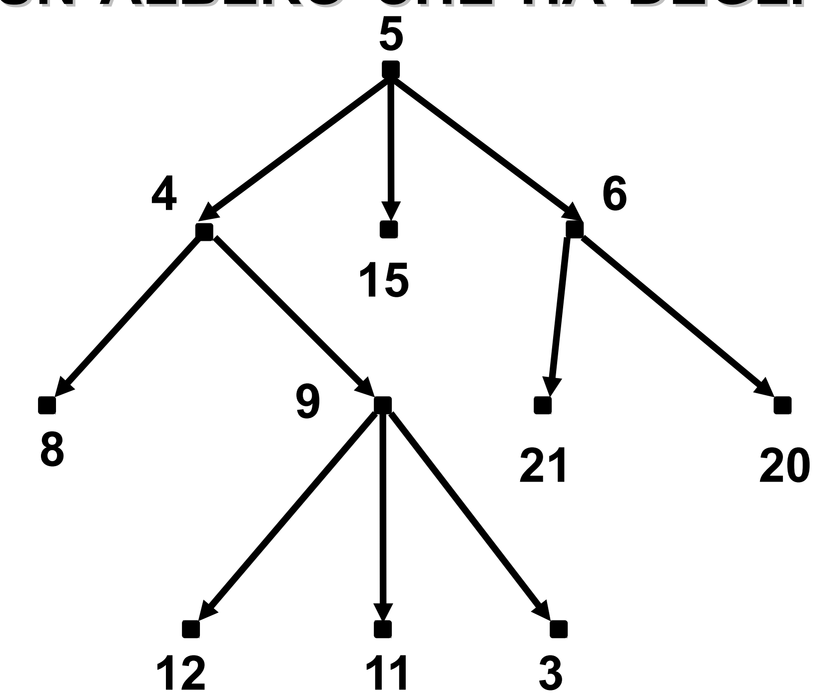
VISITA DI ALBERI

CONSISTE NEL PIANIFICARE E SEGUIRE UNA "ROTTA" CHE CONSENTA DI ESAMINARE OGNI NODO DELL'ALBERO ESATTAMENTE UNA VOLTA.
ESISTONO MODI DIVERSI PER EFFETTUARE UNA VISITA CORRISPONDENTI ALL'*ORDINE* CON CUI SI INTENDE SEGUIRE LA STRUTTURA.

SIA T UN ALBERO NON VUOTO DI RADICE r. SE r NON E' FOGLIA ED HA k (k>0) FIGLI, SIANO T_1, T_2, …, T_k I SOTTOALBERI DI T AVENTI COME RADICI I FIGLI DI r. GLI ORDINI DI VISITA SONO:

- ***PREVISITA*** (*PREORDINE*): CONSISTE NELL'ESAMINARE r E POI, NELL'ORDINE, EFFETTUARE LA PREVISITA DI T_1, T_2, …, T_k;
- ***POSTVISITA*** (*POSTORDINE*): CONSISTE NEL FARE, NELL'ORDINE, PRIMA LA POSTVISITA DI T_1, T_2, …, T_k E POI NELL'ESAMINARE LA RADICE r ;
- ***INVISITA*** (*ORD. SIMMETRICO*): CONSISTE NEL FARE, NELL'ORDINE LA INVISITA DI T_1, T_2, …, T_i, NELL'ESAMINARE r, E POI EFFETTUARE, NELL'ORDINE, LA INVISITA DI T_{i+1}, …, T_k, PER UN PREFISSATO $i \geq 1$.

ESEMPIO: **SIA UN ALBERO CHE HA DEGLI INTERI NEI NODI:**



LA VISITA IN PREORDINE HA L'EFFETTO DI VISITARE I NODI SECONDO LA SEQUENZA:

5 4 8 9 12 11 3 15 6 21 20

LA VISITA IN POSTORDINE PRODUCE:

8 12 11 3 9 4 15 21 20 6 5

LA INVISITA (i=1) PRODUCE:

8 4 12 9 11 3 5 15 21 6 20

DIAMO UNA REALIZZAZIONE IN PASCAL

```
procedure PREVISITA(var T:albero; U:nodo);
var C:nodo;
begin
    {esamina nodo U};        (1)
    if not FOGLIA(U,T) then  (2)
        begin
        C:=PRIMOFIGLIO(U,T);
        while not ULTIMOFRATELLO(C,T) do
            begin
            PREVISITA(T,C);
            C:=SUCCFRATELLO(C,T)
            end;
        PREVISITA(T,C)
        end;
end;
```

SCAMBIANDO L'ORDINE DELLE ISTRUZIONI (1) E (2) SI OTTIENE LA POSTVISITA.

DIAMO ORA LA INVISITA (PER i=1):

```
procedure INVISITA(var T:albero; U:nodo);
var C:nodo;
begin
    if FOGLIA(U,T) then
        {esamina nodo U};
    else
        begin
        C:=PRIMOFIGLIO(U,T);
        INVISITA(T,C);
        {esamina nodo U};
        while not ULTIMOFRATELLO(C,T) do
            begin
            C:=SUCCFRATELLO(C,T);
            INVISITA(T,C)
            end
        end
end;
```

EQUIVALENZA DI ALBERI N-ARI E BINARI

È EVIDENTE CHE SI TRATTA DI UNA ***EQUIVALENZA AI FINI DELLA PRE-VISITA***. E' SEMPRE POSSIBILE RAPPRESENTARE UN ALBERO N-ARIO ORDINATO **T** CON UN ALBERO BINARIO **B** AVENTE GLI STESSI NODI E LA STESSA RADICE: IN B OGNI NODO HA COME FIGLIO SINISTRO IL PRIMO FIGLIO IN T E COME FIGLIO DESTRO IL FRATELLO SUCCESSIVO IN T.

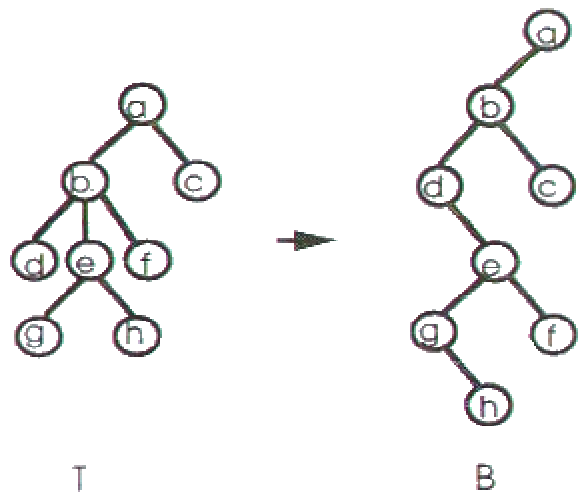


Figura 4.12: Rappresentazione di un albero ordinato T con un albero binario B.

È FACILE NOTARE CHE LE SEQUENZE DI NODI ESAMINATI SU T E SU B COINCIDONO SE T E B SONO VISITATI IN PREVISITA.

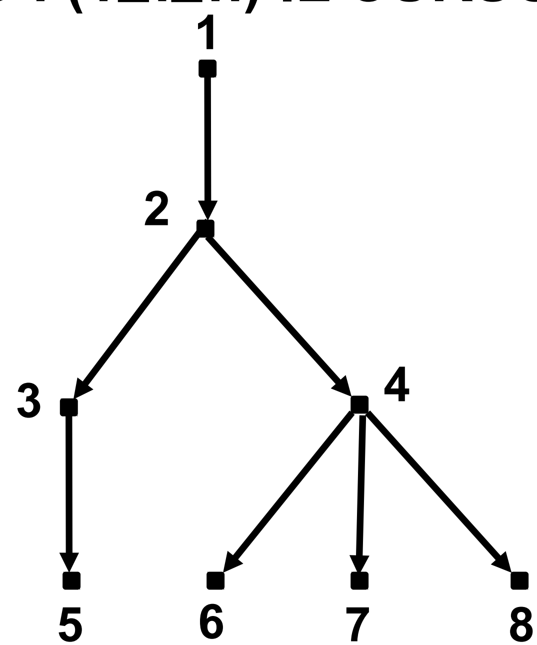
OLTRE ALLE VISITE, UN'ALTRA FUNZIONE UTILE E' LA DI PROFONDITA' DI UN ALBERO INTESA COME IL MASSIMO LIVELLO DELLE FOGLIE.

```
function MAXPROFONDITA(U:nodo; var T:albero):integer;
var V:nodo;
    MAX, CORR:integer;
begin
    if FOGLIA(U,T) then
        MAXPROFONDITA:=0
    else
        begin
        V:=PRIMOFIGLIO(U,T);
        MAX:=MAXPROFONDITA(V,T);
        repeat
            V:=SUCCFRATELLO(V,T);
            CORR:=MAXPROFONDITA(V,T);
            if MAX≤CORR then MAX:=CORR
        until ULTIMOFRATELLO(V,T);
        MAXPROFONDITA:=MAX+1
        end
end;
```

VA CHIAMATA COME MAXPROFONDITA(RADICE(T),T).

RAPPRESENTAZIONE CON VETTORE DI PADRI

IMMAGINANDO DI NUMERARE I NODI DI T DA 1 A n, LA PIÙ SEMPLICE REALIZZAZIONE (SEQUENZIALE) CONSISTE NELL'USARE UN VETTORE CHE CONTIENE, PER OGNI NODO i ($1 \leq i \leq n$) IL CURSORE AL PADRE.



1	0
2	1
3	2
4	2
5	3
6	4
7	4
8	4

È FACILE, COSÌ, VISITARE I NODI LUNGO PERCORSI CHE VANNO DA FOGLIE A RADICE. È, INVECE, PIÙ COMPLESSO INSERIRE E CANCELLARE SOTTOALBERI.

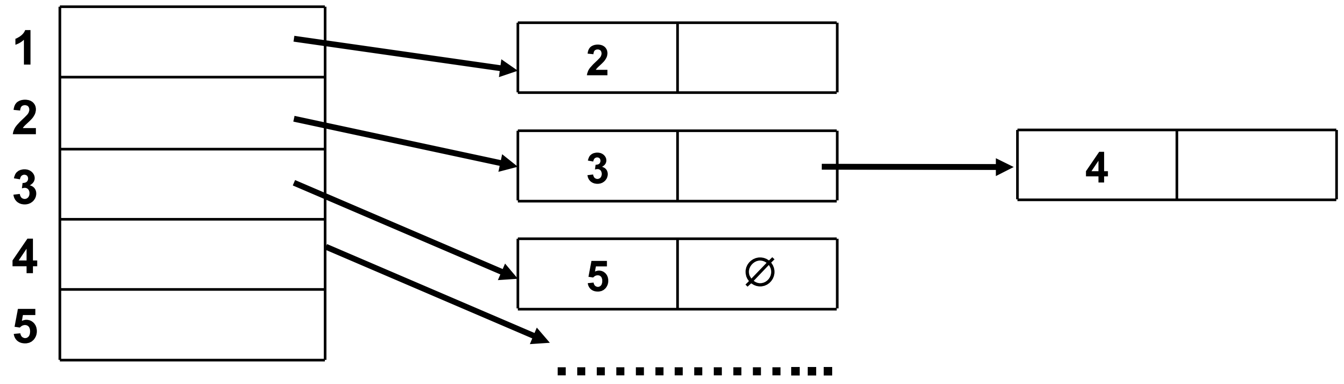
UNA VARIANTE MOLTO USATA, POICHÉ NON SI PUÒ ASSOCIARE AD OGNI ELEMENTO UN NUMERO DI PUNTATORI UGUALE AL MASSIMO DEI FIGLI, E'

LA RAPPRESENTAZIONE ATTRAVERSO LISTE DI FIGLI

COMPRENDE:

- **IL VETTORE DEI NODI, IN CUI, OLTRE ALLE EVENTUALI ETICHETTE DEI NODI, SI MEMORIZZA IL RIFERIMENTO INIZIALE DI UNA LISTA ASSOCIATA AD OGNI NODO;**
- **UNA LISTA PER OGNI NODO, DETTA *LISTA DEI FIGLI*. LA LISTA ASSOCIATA AL GENERICO NODO i CONTIENE TANTI ELEMENTI QUANTI SONO I SUCCESSORI DI i; CIASCUN ELEMENTO È IL RIFERIMENTO AD UNO DEI SUCCESSORI.**

PER L'ESEMPIO PRECEDENTE SI AVREBBE:

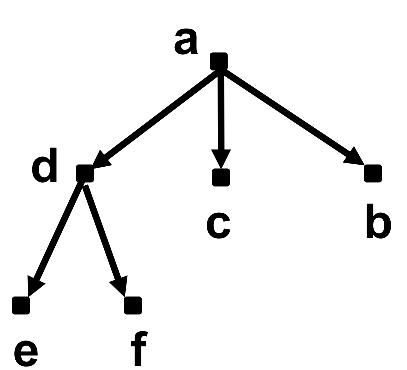


RAPPRESENTAZIONE MEDIANTE LISTA PRIMOFIGLIO/FRATELLO

PREVEDE LA GESTIONE DI UNA LISTA E QUESTO PUÒ ESSERE FATTO IMPONENDO CHE TUTTI GLI ALBERI *CONDIVIDANO UN’AREA COMUNE* (AD ESEMPIO UN VETTORE NELLA REALIZZAZIONE CON CURSORI) E CHE *OGNI CELLA CONTENGA ESATTAMENTE DUE CURSORI: UNO AL PRIMOFIGLIO ED UNO AL FRATELLO SUCCESSIVO*. LA REALIZZAZIONE È SIMILE A QUELLA PROPOSTA PER GLI ALBERI BINARI CON L’UNICA DIFFERENZA CHE IL CURSORE NEL TERZO CAMPO PUNTA AL FRATELLO. NATURALMENTE E’ POSSIBILE ANCHE PREVEDERE UN CURSORE AL *GENITORE*:

INIZIO
4

	FIGLIO	NODO	FRATELLO
1	0	e	2
2	0	f	0
3	0	c	5
4	7	a	0
5	0	b	0
6			
7	1	d	3
8			

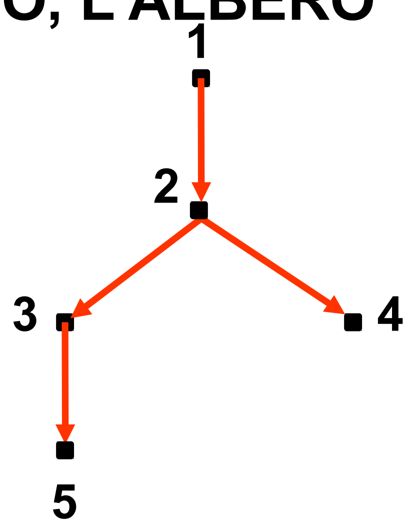


LA RAPPRESENTAZIONE COLLEGATA MEDIANTE LISTA DINAMICA

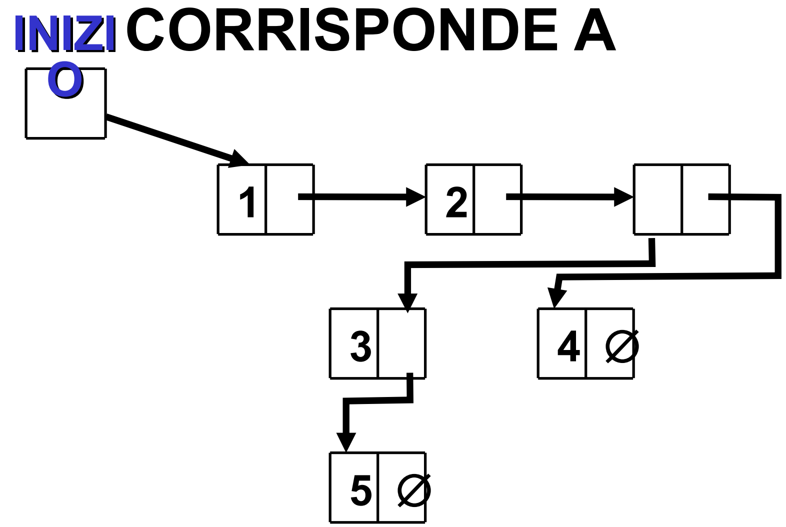
DA UN PUNTO DI VISTA FORMALE L'ALBERO N-ARIO PUÒ ESSERE RAPPRESENTATO MEDIANTE LISTA SECONDO LE SEGUENTI REGOLE:

- **SE L'ALBERO È VUOTO LA LISTA CHE LO RAPPRESENTA È VUOTA;**
- **ALTRIMENTI, L'ALBERO È COMPOSTO DA UNA RADICE E DA k SOTTOALBERI T_1, T_2, …, T_k E LA LISTA È FATTA DA k+1 ELEMENTI: IL PRIMO RAPPRESENTA LA RADICE, MENTRE GLI ALTRI SONO GLI ALBERI T_1, T_2, …, T_k (CON $k \geq 0$);**

AD ESEMPIO, L'ALBERO



CORRISPONDE A

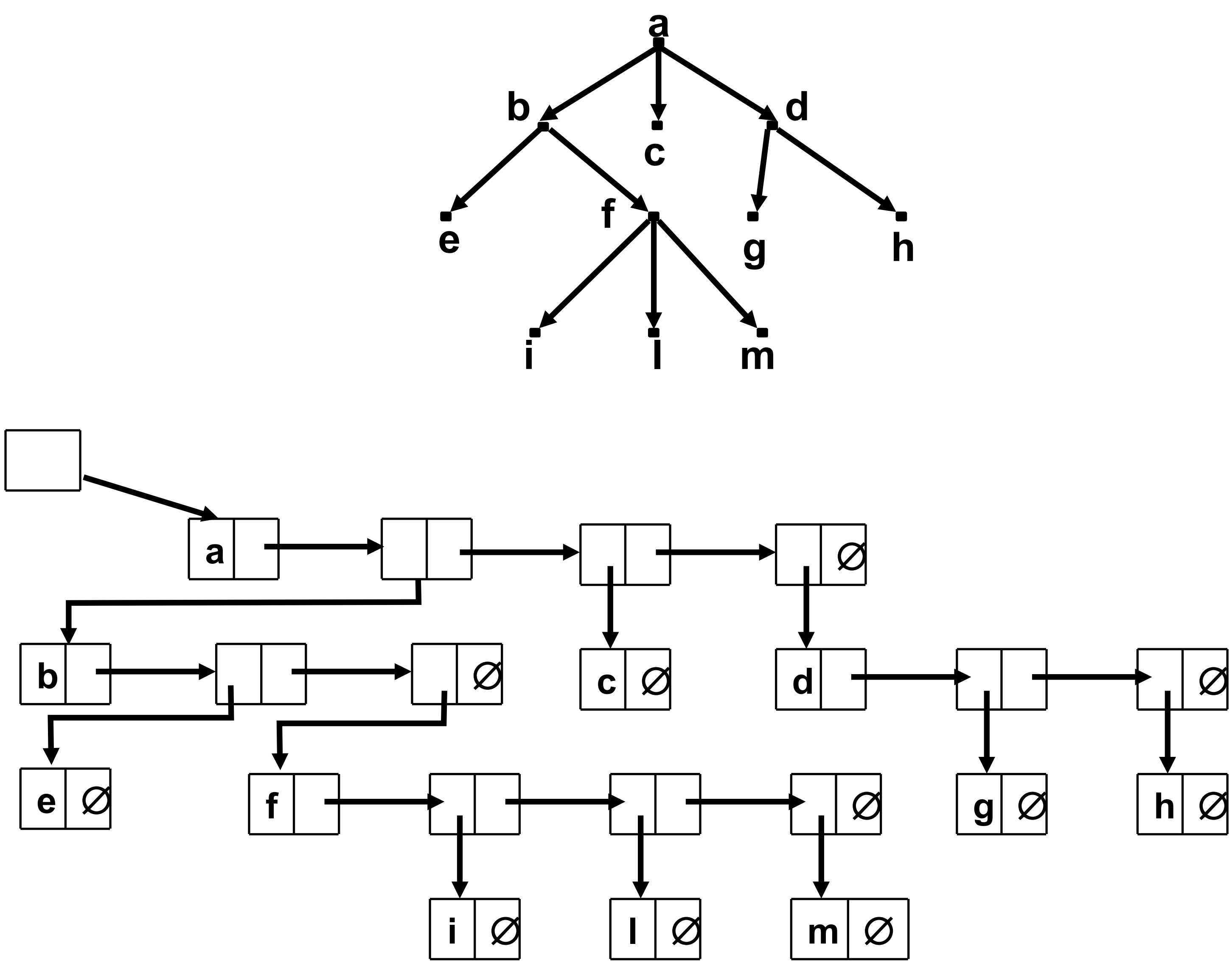


LA RAPPRESENTAZIONE COLLEGATA MEDIANTE LISTA DINAMICA

LA RAPPRESENTAZIONE CON LISTA DINAMICA È COMUNQUE COMPLESSA: IN GENERALE, LA RADICE DELL’ALBERO VIENE MEMORIZZATA NEL PRIMO ELEMENTO DELLA LISTA CHE CONTIENE IL RIFERIMENTO AD UNA LISTA DI ELEMENTI, UNO PER OGNI SOTTOALBERO. CIASCUNO DI QUESTI ELEMENTI CONTIENE, A SUA VOLTA, IL RIFERIMENTO INIZIALE ALLA LISTA CHE RAPPRESENTA IL CORRISPONDENTE SOTTOALBERO.

UNA POSSIBILE REALIZZAZIONE PREVEDE RECORD E PUNTATORI, MA IL RECORD VA INTESO CON VARIANTI: PER ESEMPIO, SI PUO’ PREVEDERE UN RECORD CON TRE CAMPI, UNO PER LA PARTE INFORMAZIONE E DUE PER I PUNTATORI. PER OGNI RECORD SARÀ’ SEMPRE SIGNIFICATIVO UNO DEI CAMPI PUNTATORE, MA QUANDO L’ATOMO RAPPRESENTA UN NODO EFFETTIVO DELL’ALBERO SARA’ UTILIZZATA L’ETICHETTA E UN PUNTATORE, QUANDO RAPPRESENTA UN ATOMO “DI SERVIZIO” SARANNO UTILIZZATI DUE PUNTATORI.

LA RAPPRESENTAZIONE COLLEGATA MEDIANTE LISTA DINAMICA



REALIZZAZIONE DI MFSET

COME E' NOTO UN MFSET È UNA PARTIZIONE DI UN INSIEME FINITO IN SOTTOINSIEMI DISGIUNTI DETTI COMPONENTI.

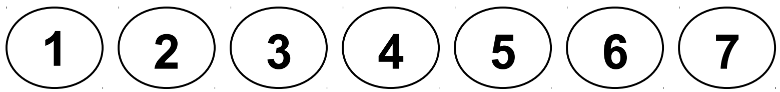
È POSSIBILE RAPPRESENTARLO MEDIANTE:

UNA FORESTA DI ALBERI RADICATI

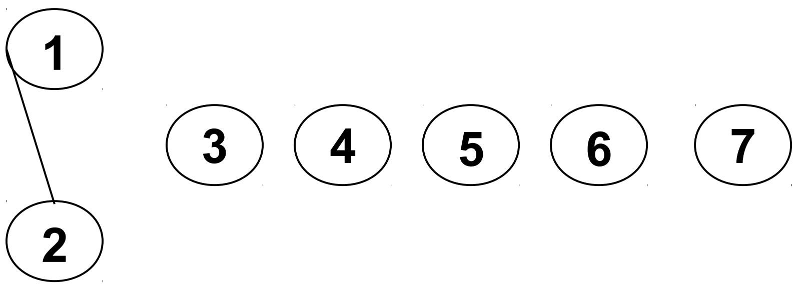
IN CUI CIASCUN ALBERO RAPPRESENTA UNA COMPONENTE.
LE COMPONENTI INIZIALI DI MFSET SONO I NODI. ATTRAVERSO OPERAZIONI SUCCESSIVE DI FONDI E TROVA SI CREA LA STRUTTURA.

L'OPERATORE FONDI COMBINA DUE ALBERI NELLO STESSO ALBERO. SI REALIZZA IMPONENDO CHE UNA DELLE DUE RADICI DIVENTI NUOVO FIGLIO DELL'ALTRA.

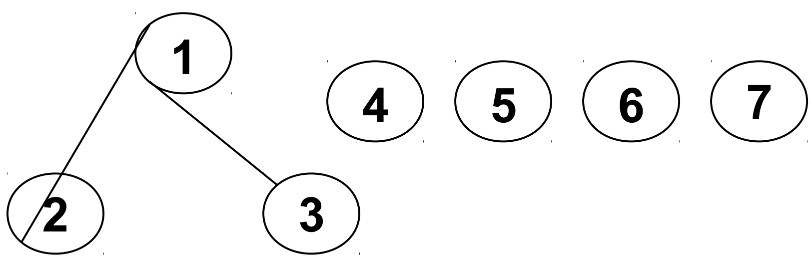
L'OPERATORE TROVA VERIFICA SE DUE ELEMENTI SONO NEL MEDESIMO ALBERO. SI REALIZZA ACCEDENDO AI NODI CONTENENTI GLI ELEMENTI E RISALENDO DA TALI NODI, ATTRAVERSO I PADRI, FINO AD ARRIVARE ALLE RADICI.



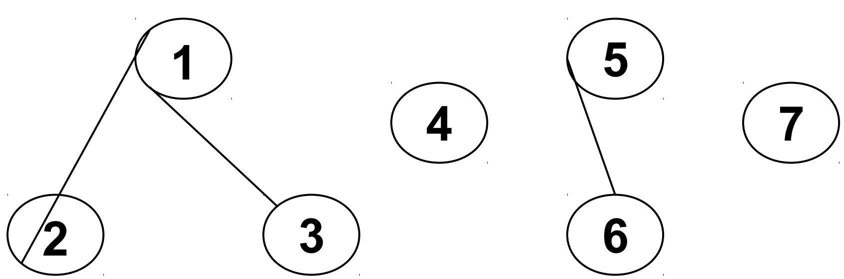
FONDI (1,2,S)



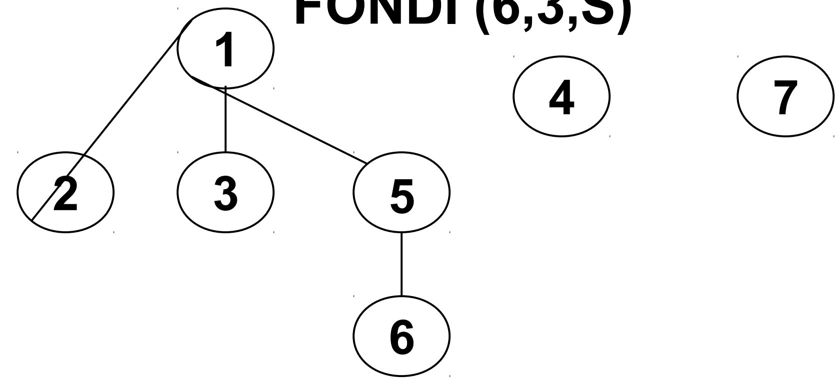
FONDI (1,3,S)



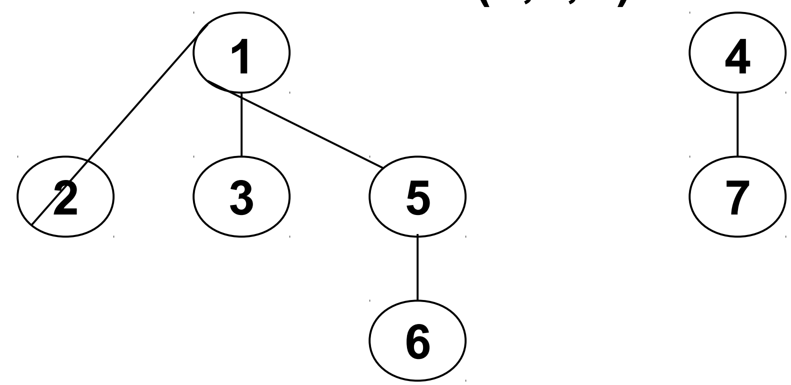
FONDI (5,6,S)



FONDI (6,3,S)



FONDI (4,7,S)



FONDI (3,7,S)