Corso di Laurea in INFORMATICA

Algoritmi e Strutture Dati

MODULO 7

STRUTTURE NON LINEARI

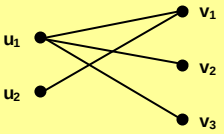
Generalità su grafi e alberi. Alberi binari: specifiche sintattiche e semantiche. Realizzazioni. Visita di alberi binari.

Questi lucidi sono stati preparati da Oronzo Altamura, Maria Francesca Costabile e Floriana Esposito, Università degli Studi di Bari, per uso didattico. Essi contengono materiale originale di proprietà dell'Università degli Studi di Bari e/o figure di proprietà di altri autori, società e organizzazioni di cui e' riportato il riferimento. Tutto o parte del materiale può essere fotocopiato per uso personale o didattico ma non può essere distribuito per uso commerciale. Qualunque altro uso richiede una specifica autorizzazione da parte dell'Università degli Studi di Bari e degli altri autori coinvolti.

GRAFI E ALBERI

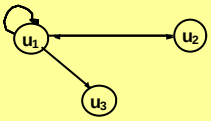
GENERALITA'

RICORDANDO CHE UNA RELAZIONE TRA DUE INSIEMI U E V E' UN SOTTOINSIEME A DI UxV, SI PUO' DARE UNA DESCRIZIONE DI A IN FORMA DIAGRAMMATICA SCRIVENDO TUTTI GLI ELEMENTI DI U, TUTTI GLI ELEMENTI DI V E CONGIUNGENDOLI. QUESTA RAPPRESENTAZIONE E' DETTA GRAFO (BIPARTITO).

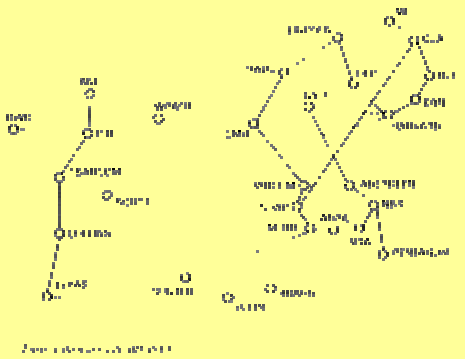


SE U • V ▶ A Í UxU. IN TAL CASO GLI ELEMENTI DI U VENGONO SCRITTI UNA SOLA VOLTA E VIENE SEGNATA UNA FRECCIA SULLA CONGIUNZIONE DA u_i A u_j PER QUELLE COPPIE $<u_i, u_j> \in A$. SI PARLA IN QUESTO CASO DI GRAFO ORIENTATO.

GLI ELEMENTI $u \in U$ SONO DETTI NODI O VERTICI DEL GRAFO ORIENTATO. LA LINEA DI CONGIUNZIONE E' DETTA ARCO.



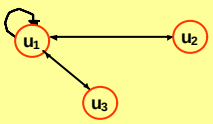
NEL CASO LE COPPIE $<u_i, u_j>$ SIANO CONGIUNTE TANTO ATTRAVERSO L'ARCO (i,j) QUANTO ATTRAVERSO L'ARCO (j,i) SI POTRA' UTILIZZARE UNA UNICA CONNESSIONE SENZA FRECCIA: L'ARCO INCIDE SUI DUE NODI E SI PARLA DI GRAFO NON ORIENTATO. GRAFI NON ORIENTATI SONO USATI PER RAPPRESENTARE RELAZIONI SIMMETRICHE TRA OGGETTI.



GRAFI ORIENTATI: DEFINIZIONI

UN GRAFO ORIENTATO G E' UNA COPPIA <N,A> DOVE N E' UN INSIEME FINITO NON VUOTO (INSIEME DI NODI) E A Í NxN E' UN INSIEME FINITO DI COPPIE ORDINATE DI NODI, DETTI ARCHI (O SPIGOLI O LINEE). SE $<u_i, u_j> \in A$ NEL GRAFO VI E' UN ARCO DA u_i AD u_j.

NELL'ESEMPIO $N = \{u_1,u_2,u_3\}$, $A = \{(u_1,u_1),(u_1,u_2),(u_2,u_1),(u_1,u_3)\}$.



GRAFI ORIENTATI: DEFINIZIONI

IN UN GRAFO ORIENTATO G UN CAMMINO E' UNA SEQUENZA DI NODI $u_0, u_1, \ldots, u_k$ TALI CHE

$(u_i, u_{i+1}) \in A$, PER $i = 0, 1, 2, \ldots, k-1$.

IL CAMMINO PARTE DAL NODO u_0, ATTRAVERSA I NODI $u_1, \ldots, u_{k-1}$, ARRIVA AL NODO u_k, ED HA LUNGHEZZA UGUALE A k.

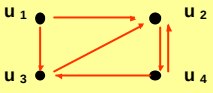
SE NON CI SONO NODI RIPETUTI IL CAMMINO E' SEMPLICE ($u_i \neq u_j$ PER $0 \leq j \leq k$).

SE $u_0 = u_k$ IL CAMMINO E' CHIUSO.

UN CAMMINO SIA SEMPLICE CHE CHIUSO E' UN CICLO.

UN GRAFO E' DETTO COMPLETO SE PER OGNI COPPIA DI NODI $u_i, u_j \in N$ ESISTE UN ARCO CHE VA DA u_i AD u_j, (A = NxN).

DEFINIREMO **GRAFO CONNESSO** UN GRAFO G = <N,A> IN CUI, DATI u E v Î N ESISTE UN CAMMINO DA u A v O UN CAMMINO DA v AD u. G E' DETTO **FORTEMENTE CONNESSO** SE PER OGNI COPPIA DI NODI u E v ESISTE ALMENO UN CAMMINO DA u A v ED ALMENO UN CAMMINO DA v AD u.

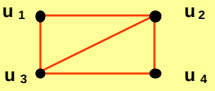


GRAFO ORIENTATO MA NON FORTEMENTE CONNESSO (NON ESISTE UN CAMMINO DA u_4 A u_1)

A CIASCUN NODO DEL GRAFO E' POSSIBILE ASSOCIARE UN'INFORMAZIONE DETTA **ETICHETTA (LABEL)** DEL NODO.

GRAFI NON ORIENTATI: DEFINIZIONI

DEFINIREMO **GRAFO NON ORIENTATO** UN GRAFO G = <N,A> NEL QUALE GLI ARCHI SONO FORMATI DA COPPIE NON ORDINATE. MENTRE IN UN GRAFO ORIENTATO (u_i,u_j) e (u_j,u_i) INDICANO DUE ARCHI DISTINTI, IN UN GRAFO NON ORIENTATO INDICANO LO STESSO ARCO CHE INCIDE SUI DUE NODI.



I NODI CONGIUNTI DA UN ARCO SONO **ADIACENTI**. NELL'ESEMPIO I NODI u_1 E u_3 SONO ADIACENTI MA u_1 E u_4 NON LO SONO. ANCHE NEI GRAFI NON ORIENTATI TROVIAMO NOZIONI ANALOGHE A QUELLE DI CAMMINO (**CATENA**) E DI CICLO (**CIRCUITO**).

IL GRAFO E' UNA STRUTTURA DATI DI GRANDE GENERALITA' ALLA QUALE SI POSSONO RICONDURRE STRUTTURE PIU' SEMPLICI: LE LISTE POSSONO ESSERE CONSIDERATE UN CASO PARTICOLARE DI GRAFO, COME PURE GLI ALBERI UTILI PER RAPPRESENTARE GERARCHIE.

AD ESEMPIO, L'ORGANIZZAZIONE DI UN INDICE:

0. TIPI DI DATO E STRUTTURE DATI
- *0.1 STRUTTURE DI DATI: SPECIFICHE REALIZZAZIONI*
- *0.2 RAPPRESENTAZIONE IN MEMORIA*

1. LISTE
- *1.1 REALIZZAZIONE CON PUNTATORI*
- *1.2 REALIZZAZIONE CON CURSORI*
- *1.3 REALIZZAZIONE CON DOPPI PUNTATORI*

IN QUESTA ORGANIZZAZIONE OGNI ARGOMENTO PRINCIPALE HA DIVERSI ARGOMENTI SECONDARI, OGNUNO DEI QUALI PUO' DIVIDERSI IN SOTTOARGOMENTI E COSI' VIA.

IN GENERALE

L'ALBERO E' UNA **STRUTTURA INFORMATIVA FONDAMENTALE** UTILE PER RAPPRESENTARE:

- PARTIZIONI SUCCESSIVE DI UN INSIEME IN SOTTOINSIEMI DISGIUNTI
- ORGANIZZAZIONI GERARCHICHE DI DATI
- PROCEDIMENTI DECISIONALI ENUMERATIVI

ALBERI

UN PARTICOLARE TIPO DI GRAFO E' L'ALBERO, DEFINITO MATEMATICAMENTE CON UNA COPPIA

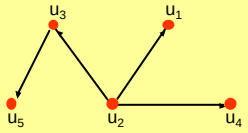
T = (N,A)

DOVE N E' UN INSIEME FINITO DI NODI ED A E' UN INSIEME DI COPPIE NON ORDINATE (**ALBERO LIBERO**) TALI CHE :

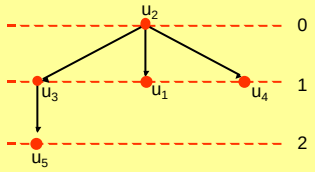
a) IL NUMERO DI ARCHI E' UGUALE AL NUMERO DI NODI MENO UNO |A| = |N| - 1

b) T E' CONNESSO, OVVERO PER OGNI COPPIA DI NODI u E v IN N, ESISTE UNA SEQUENZA DI NODI DISTINTI $u_0, u_1, \ldots, u_k$ TALI CHE $u = u_0$, $v = u_k$ E LA COPPIA $< u_i, u_{i+1} >$ E' UN ARCO DI A, PER i = 0, 1, …, k-1.

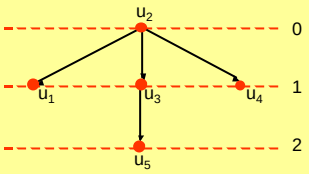
UN **ALBERO RADICATO** E' OTTENUTO DA UN ALBERO LIBERO DESIGNANDO ARBIRARIAMENTE UN NODO r COME **RADICE** E ORDINANDO I NODI PER **LIVELLI**.



ALBERO NON RADICATO



ALBERO RADICATO



ALBERO RADICATO ORDINATO

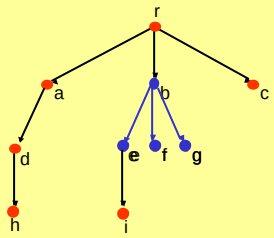
LA RADICE r E' A LIVELLO 0 E TUTTI I NODI u, "t < u, r > Î A, SONO FIGLI DI r E STANNO A LIVELLO 1 (r E' PADRE). NODI CON LO STESSO PADRE SONO FRATELLI. NODI TERMINALI SENZA FIGLI SONO DETTI FOGLIE.

UN ALBERO ORDINATO E' OTTENUTO DA UNO RADICATO STABILENDO UN ORDINAMENTO TRA NODI ALLO STESSO LIVELLO.

SE T E' UN ALBERO ED n E m SONO NODI DI T SI DICE CHE:

- m E' DISCENDENTE DI n SE n E' ANTENATO DI m CIOE' SE n=m (ANTENATO PROPRIO) OPPURE SE n E' GENITORE DI UN ANTENATO DI m
- UN NODO E' NODO INTERNO SE NON E' FOGLIA
- UNA LINEA DI T CONNETTE DUE NODI, UNO DEI QUALI E' GENITORE DELL'ALTRO
- UN CAMMINO IN T E' LA SEQUENZA DI LINEE CHE UNISCE DUE NODI, UNO DEI QUALI E' ANTENATO DELL'ALTRO: LA LUNGHEZZA DI UN CAMMINO E' COSTITUITA DAL NUMERO DI LINEE CHE LO COMPONGONO
- LA ALTEZZA DI UN NODO E' LA LUNGHEZZA DEL CAMMINO PIU' LUNGO DA QUEL NODO AD UNA FOGLIA
- LA PROFONDITA' DI UN NODO E' LA LUNGHEZZA DEL CAMMINO DALLA RADICE A QUEL NODO

DEFINIAMO ALBERO DI ORDINE K UN ALBERO IN CUI OGNI NODO HA AL MASSIMO K FIGLI



ALBERO TERNARIO

IN UN ALBERO VALGONO LE SEQUENTI PROPRIETA':

- UN ALBERO E' UN GRAFO ACICLICO, IN CUI PER OGNI NODO C'E' UN SOLO ARCO ENTRANTE (TRANNE CHE PER LA RADICE CHE NON NE HA NESSUNO)
- UN ALBERO E' UN GRAFO DEBOLMENTE CONNESSO
- SE ESISTE UN CAMMINO CHE VA DA UN NODO u AD UN ALTRO NODO v, TALE CAMMINO E' UNICO
- IN UN ALBERO ESISTE UN SOLO CAMMINO CHE VA DALLA RADICE A QUALUNQUE ALTRO NODO
- TUTTI I NODI DI UN ALBERO T (TRANNE r) POSSONO ESSERE RIPARTITI IN INSIEMI DISGIUNTI CIASCUNO DEI QUALI INDIVIDUA UN ALBERO (DATO UN NODO u, I SUOI DISCENDENTI COSTITUISCONO UN ALBERO DETTO SOTTOALBERO DI RADICE u)

LA NATURA RICORSIVA DEGLI ALBERI

UN ALBERO PUO' ESSERE DEFINITO RICORSIVAMENTE

- UN ALBERO E' UN INSIEME DI NODI AI QUALI SONO ASSOCIATE DELLE INFORMAZIONI
- TRA I NODI ESISTE UN NODO PARTICOLARE CHE E' LA RADICE (LIVELLO 0)
- GLI ALTRI NODI SONO PARTIZIONATI IN SOTTOINSIEMI CHE SONO A LORO VOLTA ALBERI (LIVELLI SUCCESSIVI)

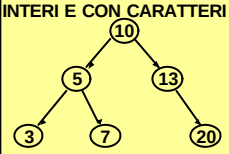
VALE A DIRE UN ALBERO E'

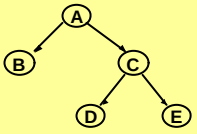
VUOTO O COSTITUITO DA UN SOLO NODO (DETTO RADICE)

OPPURE E' UNA RADICE CUI SONO CONNESSI ALTRI ALBERI

ALBERI BINARI

SONO PARTICOLARI ALBERI ORDINATI IN CUI OGNI NODO HA AL PIU' DUE FIGLI E SI FA SEMPRE DISTINZIONE TRA IL FIGLIO SINISTRO, CHE VIENE PRIMA NELL'ORDINAMENTO E IL FIGLIO DESTRO. NELL'ESEMPIO GLI ALBERI SONO ETICHETTATI CON INTERI E CON CARATTERI





DEFINIZIONE:

UN ALBERO BINARIO E' UN GRAFO ORIENTATO CHE O E' VUOTO O E' COSTITUITO DA UN SOLO NODO O E' FORMATO DA UN NODO N (DETTO RADICE) E DA DUE SOTTOALBERI BINARI, CHE VENGONO CHIAMATI RISPETTIVAMENTE SOTTOALBERO (O FIGLIO) SINISTRO E SOTTOALBERO (O FIGLIO) DESTRO

LA SPECIFICA SINTATTICA

TIPI: ALBEROBIN, BOOLEANO, NODO

CREABINALBERO : ( ) → ALBEROBIN

BINALBEROVUOTO : (ALBEROBIN) → BOOLEANO

BINRADICE : (ALBEROBIN) → NODO

BINPADRE : (NODO,ALBEROBIN) → NODO

FIGLIOSINISTRO : (NODO,ALBEROBIN) → NODO

FIGLIODESTRO : (NODO,ALBEROBIN) → NODO

SINISTROVUOTO : (NODO,ALBEROBIN) → BOOLEANO

DESTROVUOTO : (NODO,ALBEROBIN) → BOOLEANO

COSTRBINALBERO : (ALBEROBIN,ALBEROBIN) → ALBEROBIN

CANCSOTTOBINALBERO : (NODO,ALBEROBIN) → ALBEROBIN

19

LA SPECIFICA SEMANTICA

TIPI: ALBEROBIN: insieme degli alberi binari T=(N,A), nei quali ad ogni nodo è associato un LIVELLO, BOOLEANO, NODO

CREABINALBERO = T'

POST: T' = Λ

BINALBEROVUOTO(T) = b

POST: b=VERO SE T = Λ; b=FALSO ALTRIMENTI

BINRADICE(T) = u

PRE: T ≠ Λ

POST: u → RADICE DI T → LIVELLO(u) = 0

BINPADRE(u,T) = v

PRE: T ≠ Λ, u ∈ N, LIVELLO(u) > 0

POST: v E' PADRE DI u → (u,v) ∈ A → LIVELLO(u)=LIVELLO(v)+1

20

FIGLIOSINISTRO(u,T) = v

PRE: T ≠ Λ, u ∈ N, u HA UN FIGLIO SINISTRO

POST: v E' IL FIGLIO SINISTRO DI u IN T

FIGLIODESTRO(u,T) = V

PRE: T ≠ Λ, u ∈ N, u HA UN FIGLIO DESTRO

POST: v E' IL FIGLIO DESTRO DI u IN T

SINISTROVUOTO(u,T) = b

PRE: T ≠ Λ, u ∈ N

POST: b=VERO SE u NON HA UN FIGLIO SINISTRO

b=FALSO ALTRIMENTI

21

DESTROVUOTO(u,T) = b

PRE: T ≠ Λ, u ∈ N

POST: b=VERO SE u NON HA UN FIGLIO DESTRO

b=FALSO ALTRIMENTI

COSTRBINALBERO(T,T') = T''

POST: T'' SI OTTIENE DA T E DA T' INTRODUCENDO AUTOMATICAMENTE UN NUOVO NODO r'' (RADICE DI T'') CHE AVRA' COME SOTTOALBERO SINISTRO T E SOTTOALBERO DESTRO T' (*SE T = Λ E T' = Λ, L'OPERATORE INSERISCE LA SOLA RADICE r''; SE T = Λ, r'' NON HA FIGLIO SINISTRO; SE T' = Λ, r'' NON HA FIGLIO DESTRO*)

22

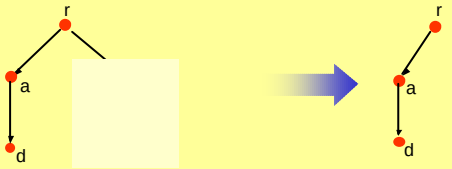
CANCSOTTOBINALBERO(u,T) = T'

PRE: T ≠ Λ, u ∈ N

POST: T' E' OTTENUTO DA T ELIMINANDO IL SOTTOALBERO DI RADICE u, CON TUTTI I SUOI DISCENDENTI

VALIDA PER ALBERI DI OGNI ORDINE AGISCE POTANDO DAL NODO u. AD ESEMPIO:

CANCSOTTOBINALBERO(b,T)



23

ANCORA DUE OPERATORI UTILI!!!

SPECIFICHE SINTATTICHE

Tipi: Va aggiunto TIPOELEM del tipo dell'etichetta

LEGGINODO: (NODO,ALBEROBIN) → TIPOELEM

SCRIVINODO: (TIPOELEM,NODO,ALBEROBIN) → ALBEROBIN

SPECIFICHE SEMANTICHE

LEGGINODO (n,T) = a

PRE: n E' UN NODO DI T, n ∈ N

POST: a E' IL VALORE ASSOCIATO AL NODO n IN T

SCRIVINODO (a,n,T) = T'

PRE: n E' UN NODO DI T, n ∈ N

POST: T' E' IL NUOVO ALBERO CORRISPONDENTE AL VECCHIO T CON IL VALORE a ASSEGNATO AL NODO n

24

L'ALGEBRA CHE ABBIAMO PRESENTATO OVVIAMENTE RAPPRESENTA UNA SCELTA PRECISA DI PROGETTO

SI E' SCELTO DI ENFATIZZARE LA NATURA RICORSIVA DEGLI ALBERI E DI COSTRUIRE L'ALBERO BINARIO DAL BASSO VERSO L'ALTO, CIOE' DAL LIVELLO DELLE FOGLIE VERSO LA RADICE.

NON SEMPRE QUESTA SCELTA E' OPPORTUNA: SOPRATTUTTO SE L'ALBERO E' USATO PER RAPPRESENTARE UN PROCESSO DECISIONALE E' PREFERIBILE UN'ALGEBRA CHE PREVEDA DI COSTRUIRE L'ALBERO DALL'ALTO VERSO IL BASSO, INSERENDO PRIMA LA RADICE E POI I NODI FIGLI VIA VIA.

IN TAL CASO, MENTRE RIMANGONO VALIDI GLI OPERATORI CREABINALBERO, BINALBEROVUOTO, BINRADICE, BINPADRE, FIGLIOSINISTRO, FIGLIODESTRO, SINISTROVUOTO, DESTROVUOTO, CANCSOTTOBINALBERO ANDREBBE SOSTITUITO L'OPERATORE DI COSTRUZIONE CON TRE OPERATORI NUOVI, UNO DEDICATO ALL'INSERIMENTO DELLA RADICE E GLI ALTRI DUE DEDICATI ALL'INSERIMENTO DEL FIGLIO SINISTRO E DEL FIGLIODESTRO.

LA SPECIFICA SINTATTICA

INSBINRADICE: (*nodo, alberobin*) ® *alberobin*
INSFIGLIOSINISTRO: (*nodo, alberobin*) ® *alberobin*
INSFIGLIODESTRO: (*nodo, alberobin*) ® *alberobin*

LA SPECIFICA SEMANTICA

INSBINRADICE(u, T) =T'
PRE: T=L
POST: T'=(N,A), N={u}, LIVELLO(u)=0, A=Ø
INSFIGLIOSINISTRO(u, T) = T'
PRE: T¹L, u Î N, SINISTROVUOTO(u)= True
POST: N'= NÈv, T' E' OTTENUTO DA T AGGIUNGENDO v COME FIGLIO SINISTRO DI u
INSFIGLIODESTRO(u, T) = T'
PRE: T¹L, u Î N, DESTROVUOTO(u)= True
POST: N'= NÈv , T' E' OTTENUTO DA T AGGIUNGENDO v COME FIGLIO DESTRO DI u

OLTRE ALLE OPERAZIONI CITATE, PER GLI ALBERI IN GENERE E PER GLI ALBERI BINARI IN PARTICOLARE, SI DEFINISCONO I COSIDDETTI

ALGORITMI DI VISITA

CIOE' ALGORITMI CHE CONSENTONO DI ANALIZZARE TUTTI I NODI DELL'ALBERO IN UN ORDINE DEFINITO.

RISULTANO PARTICOLARMENTE IMPORTANTI IN PROBLEMI PER I QUALI, AD ESEMPIO, SI DEBBA RICERCARE IN QUALE NODO O A QUALE LIVELLO E' CONTENUTO IN ETICHETTA UN VALORE DATO IN INPUT OPPURE QUANDO SI VOGLIA ESPLORARE L'ALBERO PER VERIFICARNE LA PROFONDITA'.

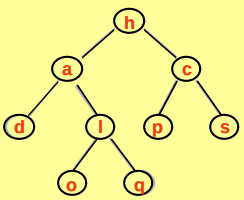
LA VISITA DI UN ALBERO CONSISTE NEL SEGUIRE UNA ROTTA DI VIAGGIO CHE CONSENTA DI ESAMINARE OGNI NODO DELL'ALBERO ESATTAMENTE UNA VOLTA.

I PIU' COMUNI ALGORITMI DI VISITA SONO TRE:

- **VISITA IN PRE-ORDINE**: SI APPLICA AD UN ALBERO NON VUOTO E RICHIEDE DAPPRIMA L'ANALISI DELLA RADICE DELL'ALBERO E, POI, LA VISITA, EFFETTUATA CON LO STESSO METODO, DEI DUE SOTTOALBERI, PRIMA IL SINISTRO, POI IL DESTRO
- **VISITA IN POST-ORDINE**: SI APPLICA AD UN ALBERO NON VUOTO E RICHIEDE DAPPRIMA LA VISITA, EFFETTUATA CON LO STESSO METODO, DEI SOTTOALBERI, PRIMA IL SINISTRO E POI IL DESTRO, E, IN SEGUITO, L'ANALISI DELLA RADICE DELL'ALBERO
- **VISITA SIMMETRICA**: RICHIEDE PRIMA LA VISITA DEL SOTTOALBERO SINISTRO (EFFETTUATA SEMPRE CON LO STESSO METODO), POI L'ANALISI DELLA RADICE, E POI LA VISITA DEL SOTTOALBERO DESTRO

ESEMPIO:

SIA UN ALBERO BINARIO CHE HA DEI CARATTERI NEI NODI



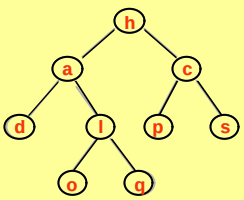
LA VISITA IN PREORDINE: h a d l o q c p s

LA VISITA IN POSTORDINE: d o q l a p s c h

LA VISITA SIMMETRICA: d a o l q h p c s

ESEMPIO:

SIA UN ALBERO BINARIO CHE HA DEI CARATTERI NEI NODI



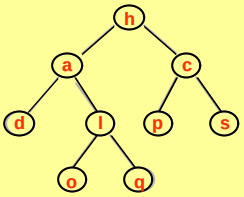
LA VISITA IN PREORDINE: h a d l o q c p s

LA VISITA IN POSTORDINE: d o q l a p s c h

LA VISITA SIMMETRICA: d a o l q h p c s

ESEMPIO:

SIA UN ALBERO BINARIO CHE HA DEI CARATTERI NEI NODI



LA VISITA IN PREORDINE: h a d l o q c p s

LA VISITA IN POSTORDINE: d o q l a p s c h

LA VISITA SIMMETRICA: d a o l q h p c s

31

LA FORMULAZIONE DEGLI ALGORITMI DI VISITA

GLI ALGORITMI SI POSSONO FACILMENTE FORMULARE IN MODO RICORSIVO. AD ESEMPIO:

VISITA IN PREORDINE L'ALBERO BINARIO T

SE L'ALBERO NON E' VUOTO

ALLORA

VISITA LA RADICE DI T

VISITA IN PREORDINE IL SOTTOALBERO SINISTRO DI T

VISITA IN PREORDINE IL SOTTOALBERO DESTRO DI T

FINE

32

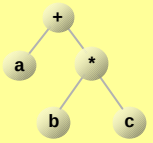
NELLA PROCEDURA SEGUENTE, LA PREVISITA SI OTTIENE ESAMINANDO IL NODO u SOLO NELL'ISTRUZIONE (1), MENTRE LA VISITA SIMMETRICA SI HA ESAMINANDO IL NODO u SOLO IN (2) E LA POSTVISITA ESAMINANDO IL NODO u SOLO IN (3).

```
PROCEDURE BINVISITA (u:NODO; VAR T:ALBERO);
BEGIN
    IF BINALBEROVUOTO(T)=FALSE
    THEN    BEGIN
(1) {ESAME ANTICIPATO DEL NODO u}
                IF NOT SINISTROVUOTO(u,T) THEN
                        BINVISITA(FIGLIOSINISTRO(u,T),T);
(2) {ESAME SIMMETRICO DEL NODO u}
                IF NOT DESTROVUOTO(u,T) THEN
                        BINVISITA(FIGLIODESTRO(u,T),T);
(3)  {ESAME DIFFERITO DEL NODO u};
            END
END
```

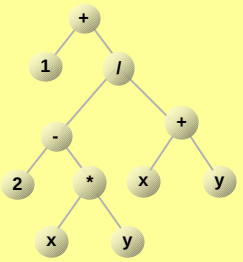
33

APPLICAZIONI: ALBERI DI ANALISI (PARSE TREE)

RAPPRESENTANO ESPRESSIONI DA VALUTARE COMINCIANDO DAL BASSO VERSO L'ALTO.



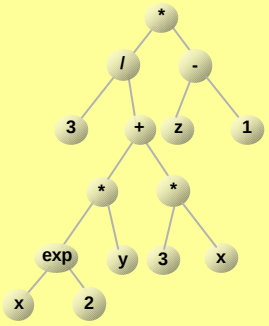
a + b * c



1 + 2- x * y
x + y

34

ESEMPIO DI ALBERO DI ANALISI (PARSE TREE)



(3/(x2y+3x))*(z-1)

35

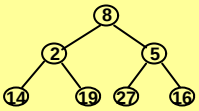
```
( * ( / ( 3 )
        ( + ( * ( exp ( x ) ( 2 ) )
              ( y )
            )
            ( + ( 3 ) ( x ) )
        )
    ( - ( z ) ( 1 ) )
)
```

(3/(x2y+3x))*(z-1)

36

LE RAPPRESENTAZIONI

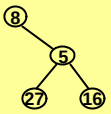
UNA POSSIBILE RAPPRESENTAZIONE DI UN ALBERO BINARIO E' QUELLA **SEQUENZIALE** MEDIANTE VETTORE. LA RADICE E' IN PRIMA POSIZIONE; PER IL GENERICO NODO p MEMORIZZATO IN POSIZIONE i, SE ESISTE IL FIGLIO SINISTRO E' MEMORIZZATO IN POSIZIONE 2*i, SE ESISTE IL FIGLIO DESTRO E' MEMORIZZATO IN POSIZIONE 2*i+1



1	8
2	2
3	5
4	14
5	19
6	27
7	16

RAPPRESENTAZIO MEDIANTE VETTORE SEQUENZIALE

SE L'ALBERO E' INCOMPLETO



1	8
2	-
3	5
4	-
5	-
6	27
7	16

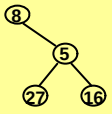
REALIZZAZIONE SEQUENZIALE

PROBLEMI

IN QUESTO CASO ALCUNE COMPONENTI DEL VETTORE NON CORRISPONDONO AD ALCUN NODO DELL'ALBERO.

QUESTO, IN CASO DI REALIZZAZIONE CON LINGUAGGI A TIPIZZAZIONE FORTE POTREBBE GENERARE PROBLEMI, NON AVENDO UN MODO DI AVVALORARE, CON UN b O CON ALTRO, ELEMENTI DI TIPO PER I QUALI DOVREMMO ESPRIMERE UN "NON DEFINITO".

LA **SOLUZIONE** E' QUELLA DI UTILIZZARE UNA RAPPRESENTAZIONE CHE ASSOCIA AD OGNI COMPONENTE DELL'ARRAY UN CAMPO DI TIPO BOOLEANO CHE VARRA' **VERO** SE NELLA COMPONENTE E' EFFETTIVAMENTE PRESENTE UN NODO DELL'ALBERO, **FALSO** ALTRIMENTI.



1	8
2	-
3	5
4	-
5	-
6	27
7	16

1	VERO	8
2	FALSO	24
3	VERO	5
4	FALSO	62
5	FALSO	3
6	VERO	27
7	VERO	16

TUTTAVIA E' IMMEDIATO VERIFICARE CHE:

- ALBERI BINARI NON COMPLETI VENGONO RAPPRESENTATI CON SPRECO DI MEMORIA,
- E' IMPOSTO UN LIMITE MASSIMO PER IL NUMERO DI NODI DELL'ALBERO
- LE OPERAZIONI DI AGGIUNTA ED ELIMINAZIONE DI NODI O DI SOTTOALBERI COMPORTANO DIVERSI AGGIORNAMENTI NELL'ARRAY.

LE RAPPRESENTAZIONI

PRIMA DI INTRODURRE LA RAPPRESENTAZIONE COLLEGATA VA FATTA QUALCHE CONSIDERAZIONE TEORICA CIRCA LA CORRISPONDENZA TRA "ALBERO BINARIO" E LA "LISTA"

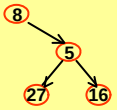
OGNI VALORE T DEL TIPO ALBERO PUO' ESSERE RAPPRESENTATO MEDIANTE UN TIPO LISTA NEL MODO SEGUENTE:

- SE T E' VUOTO, LA LISTA CHE LO RAPPRESENTA E' LA LISTA VUOTA
- SE T NON E' VUOTO, LA LISTA CHE LO RAPPRESENTA E' FORMATA DA TRE ELEMENTI:
  - IL PRIMO E' L'ATOMO CHE RAPPRESENTA LA RADICE DI T
  - IL SECONDO E' UNA LISTA CHE RAPPRESENTA, CON LO STESSO METODO, IL SOTTOALBERO SINISTRO DI T
  - IL TERZO E' UN'ALTRA LISTA CHE RAPPRESENTA IL SOTTOALBERO DESTRO DI T

POSSIAMO USARE UNA RAPPRESENTAZIONE CON PARENTESI PER RAPPRESENTARE UN ALBERO BINARIO MEDIANTE LISTA

()	ALBERO VUOTO
(a)	ALBERO COSTITUITO DALLA SOLA RADICE
(a () ())	ALBERO BINARIO COSTITUITO DA RADICE a, UN FIGLIO SINISTRO VUOTO E UN FIGLIO DESTRO VUOTO

AD ESEMPIO L'ALBERO SEGUENTE



CORRISPONDE ALLA LISTA

(8 () (5(27 () ()) (16 () ())))

LA RAPPRESENTAZIONE COLLEGATA DI UN ALBERO



UN PRIMO METODO RICHIEDE DI UTILIZZARE UN ARRAY, IN MODO CHE AD OGNI NODO DELL'ALBERO CORRISPONDA UNA COMPONENTE DELL'ARRAY IN CUI SONO MEMORIZZATE LE INFORMAZIONI (NODO, RIFERIMENTO AL FIGLIO SINISTRO, RIFERIMENTO AL FIGLIO DESTRO).

IL RIFERIMENTO E' IL VALORE DELL'INDICE IN CORRISPONDENZA DEL QUALE SI TROVA LA COMPONENTE CHE CORRISPONDE AL FIGLIO SINISTRO O DESTRO.

SE IL FIGLIO NON ESISTE IL RIFERIMENTO HA VALORE 0.

43

LA RAPPRESENTAZIONE COLLEGATA DI UN ALBERO

UN PRIMO METODO RICHIEDE DI UTILIZZARE UN ARRAY, IN MODO CHE AD OGNI NODO DELL'ALBERO CORRISPONDA UNA COMPONENTE DELL'ARRAY IN CUI SONO MEMORIZZATE LE INFORMAZIONI (NODO, RIFERIMENTO AL FIGLIO SINISTRO, RIFERIMENTO AL FIGLIO DESTRO).

IL RIFERIMENTO E' IL VALORE DELL'INDICE IN CORRISPONDENZA DEL QUALE SI TROVA LA COMPONENTE CHE CORRISPONDE AL FIGLIO SINISTRO O DESTRO.

SE IL FIGLIO NON ESISTE IL RIFERIMENTO HA VALORE 0.

44

LA RAPPRESENTAZIONE COLLEGATA DI UN ALBERO



UN PRIMO METODO RICHIEDE DI UTILIZZARE UN ARRAY, IN MODO CHE AD OGNI NODO DELL'ALBERO CORRISPONDA UNA COMPONENTE DELL'ARRAY IN CUI SONO MEMORIZZATE LE INFORMAZIONI (NODO, RIFERIMENTO AL FIGLIO SINISTRO, RIFERIMENTO AL FIGLIO DESTRO).

IL RIFERIMENTO E' IL VALORE DELL'INDICE IN CORRISPONDENZA DEL QUALE SI TROVA LA COMPONENTE CHE CORRISPONDE AL FIGLIO SINISTRO O DESTRO.

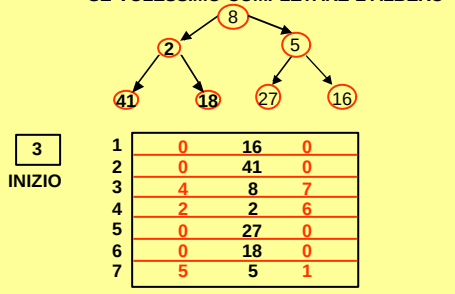
SE IL FIGLIO NON ESISTE IL RIFERIMENTO HA VALORE 0.

45

SE VOLESSIMO COMPLETARE L'ALBERO

46

SE VOLESSIMO COMPLETARE L'ALBERO

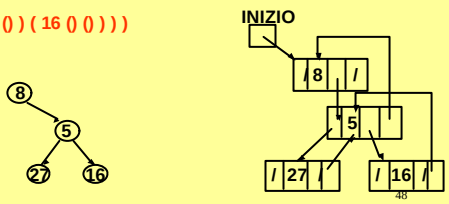


47

LA RAPPRESENTAZIONE COLLEGATA CON USO DI VARIABILI DINAMICHE

PROPRIO PERCHE' L'ALBERO BINARIO PUO' ESSERE VISTO COME UNA LISTA, OVVIAMENTE E' POSSIBILE USARE PUNTATORI INVECE CHE CURSORI E LA MANCANZA DI UN FIGLIO VIENE INDICATA COL VALORE nil NELL'APPOSITO CAMPO. PREVEDIAMO UN CAMPO PER IL FIGLIO DESTRO, UNO PER IL FIGLIO SINISTRO E, PER RAGIONI DI EFFICIENZA UN CAMPO PER IL PADRE

(8 () (5(27 () ()) (16 () ())))



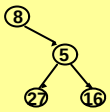
48

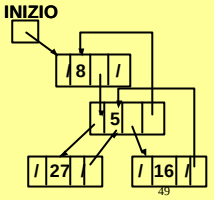
UN ESEMPIO DI REALIZZAZIONE CON USO DI PUNTATORI

COME SEMPRE USIAMO IL LINGUAGGIO PASCAL PER FORNIRE UN ESEMPIO DI REALIZZAZIONE DI ALBERO BINARIO.

UNA POSSIBILE DICHIARATIVA:

```
TYPE
BINCELLA = RECORD
            SINISTRO, DESTRO, GENITORE: ^BINCELLA;
            NOME: TIPONOME
      END;
BINALBERO = ^BINCELLA;
NODO = ^BINCELLA;
```





49

```
PROCEDURE CREABINALBERO (VAR T:BINALBERO);
      BEGIN
            T:=nil;
      END;
-----------------------------------------------------------------
FUNCTION BINALBEROVUOTO (VAR T:BINALBERO):BOOLEAN;
      BEGIN
            BINALBEROVUOTO:=(T=nil)
      END;
-----------------------------------------------------------------
FUNCTION BINRADICE (VAR T:BINALBERO):NODO;
      BEGIN
            BINRADICE:=T;
      END;
```

50

```
FUNCTION BINPADRE (u:NODO; VAR T:BINALBERO):NODO;
BEGIN
      BINPADRE:=u^.GENITORE;
END;
-----------------------------------------------------------------
FUNCTION FIGLIOSINISTRO(u:NODO; VAR T:BINALBERO):NODO;
BEGIN
      FIGLIOSINISTRO:=u^.SINISTRO;
END;
-----------------------------------------------------------------
FUNCTION FIGLIODESTRO è analoga
-----------------------------------------------------------------
FUNCTION SINISTROVUOTO(u:NODO; VAR T:BINALBERO):
                              BOOLEAN;
BEGIN
      SINISTROVUOTO:=(u^.SINISTRO= nil)
END;
```

51

```
PROCEDURE CONSTRBINALBERO (VAR T,U:BINALBERO);
VAR u:NODO;
BEGIN
NEW(u);
u^.SINISTRO:=T;
u^.DESTRO:=U;
u^.GENITORE:=nil;
IF T ¹ nil THEN T^.GENITORE:=u;
IF U ¹ nil THEN U^.GENITORE:=u;
T:=u;
END;
```

52

```
PROCEDURE CANCSOTTOBINALBERO (u:NODO; VAR
                                    T:BINALBERO);
 {utilizza una procedura interna CANC}
      PROCEDURE CANC (n:NODO);
      BEGIN
        IF (n^.SINISTRO^.SINISTRO <> nil OR
            n^.SINISTRO^.DESTRO <> nil) THEN
        CANC(n^.SINISTRO);
        IF (n^.DESTRO^.SINISTRO <> nil OR
            n^.DESTRO^.DESTRO <> nil) THEN
        CANC(n^.DESTRO);
        IF ( n^.SINISTRO <> nil) THEN DISPOSE(n^.SINISTRO);
        IF ( n^.DESTRO <> nil) THEN DISPOSE(n^.DESTRO)
      END;          {fine della procedura CANC}
```

53

```
BEGIN
      IF (u = T) THEN T:=nil;
      IF (u <> nil) THEN
            BEGIN
              CANC(u);
              DISPOSE(u)
            END
END;
```

54

```
FUNCTION LEGGINODO (n:NODO; T:BINALBERO):TIPONOME;
BEGIN
        LEGGINODO:=n^.NOME;
END;
PROCEDURE SCRIVINODO (a:TIPONOME; n:NODO; var
                                        T:BINALBERO);
BEGIN
        n^.NOME:=a;
END;
```

55

PROBLEMA: NUMERO NODI PER SOTTOALBERO

DATO UN ALBERO BINARIO T, NON VUOTO, SI MEMORIZZI NELL'ETICHETTA DI OGNI NODO u IL NUMERO DI NODI CHE SI TROVANO NEL SOTTOALBERO CON RADICE IN u.

```
procedure CONTANODI(U:nodo; var T:binalbero);
var     SOMMASIN, SOMMADES, CONTO: integer;
begin
  if (SINISTROVUOTO(U,T)=true) and (DESTROVUOTO(U,T)=true) then
            CONTO:=1
            SCRIVINODO(CONTO,U,T)
     else
        begin
          if not SINISTROVUOTO(U,T) then
             begin
               CONTANODI(FIGLIOSINISTRO(U,T),T);
               SOMMASIN:=LEGGINODO(FIGLIOSINISTRO(U,T),T);
             end
          else
             SOMMASIN:=0;
```

56

```
          if not DESTROVUOTO(U,T) then
              begin
                  CONTANODI(FIGLIODESTRO(U,T),T);
                  SOMMADES:=LEGGINODO(FIGLIODESTRO(U,T),T);
              end
           else
              SOMMADES:=0;
          CONTO:=SOMMASIN+SOMMADES+1;
          SCRIVINODO(CONTO,U,T)
       end
end;
```

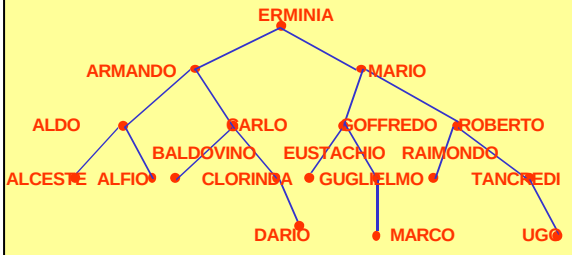
57

PROBLEMA: RICERCA BINARIA DI UN NOME IN UNA TABELLA

ALCESTE
ALDO
ALFIO
ARMANDO
BALDOVINO
CARLO
CLORINDA
DARIO
ERMINIA
EUSTACHIO
GOFFREDO
GUGLIELMO
MARCO
MARIO
RAIMONDO
ROBERTO
TANCREDI
UGO

58

IL PROCEDIMENTO DI RICERCA BINARIA DI UN NOME IN UNA TABELLA PUO' ESSERE VISUALIZZATO MEDIANTE UN ALBERO BINARIO, DETTO ALBERO BINARIO DI RICERCA.



SI PUÒ OSSERVARE CHE, PER OGNI NODO u:

- TUTTE LE CHIAVI ASSOCIATE A NODI CHE SI TROVANO NEL SOTTOALBERO SINISTRO DI UN DETERMINATO NODO INTERNO u SONO MINORI DELLA CHIAVE ASSOCIATA AL NODO u;
- TUTTE LE CHIAVI ASSOCIATE A NODI CHE SI TROVANO NEL SOTTOALBERO DESTRO DI UN DETERMINATO NODO INTERNO u SONO MAGGIORI DELLA CHIAVE ASSOCIATA AL NODO u.

59

RICERCA DELLA CHIAVE "CLORINDA"

ALDO
ALCESTE
ALFIO
ARMANDO
BALDOVINO
CARLO
CLORINDA
DARIO
ERMINIA
EUSTACHIO
GOFFREDO
GUGLIELMO
MARCO
MARIO
RAIMONDO
ROBERTO
TANCREDI
UGO

60

RICERCA BINARIA (IN UNA TABELLA)

```
procedure RICERCA_BINARIA(var A:tabella; K:chiave;
                                        var SUCCESSO:boolean);
var  MAX, MIN, MED:integer;
begin
     MAX:=N;
     MIN:=1;
     SUCCESSO:=false;
     while (MAX ³ MIN) do
         begin
            MED:=(MAX – MIN)/2;
            if (A[MED].ATTR_CHIAVE=K) then
                begin
                SUCCESSO:=true;
                end
             else
                if (A[MED].ATTR_CHIAVE>K) then
                    MAX:=MED-1
                 else
                    MIN:=MED+1
         end
end;
```

61