COMPITO SCRITTO DI INTELLIGENZA ARTIFICIALE
22 OTTOBRE 2001 (Prof. Lamma)

h. 10.30 AULA 1 - Durata 3 ore
(Punteggio 31/30; Risultati 25 Ottobre 2001;
ORALI: 25 Ottobre, 8 Novembre, 15 Novembre h. 10.30 c/o Dipartimento di Ingegneria)

Esercizio 1: (PUNTI 6)
Si traducano in logica dei predicati del primo ordine le seguenti frasi

- Esiste un bambino di nome Andrea
- Un bambino o scrive o balla, ma non fa le due attività insieme
- Quando i bambini ascoltano musica ballano
- Andrea ascolta musica al mattino

Si dimostri tramite la risoluzione che cosa fa Andrea il mattino (trovare entrambe le soluzioni).

Esercizio 2: (PUNTI 7)
Trovare un numero di due cifre decimali, compreso fra 1 e 99, tale che la somma delle due cifre sommata al numero stesso dia un secondo numero, ottenuto invertendo le cifre del primo.
Si rappresenti il problema descritto come CSP e lo si risolva con utilizzando la tecnica forward checking.
Suggerimento: si utilizzino due variabili, A0 e A1, che rappresentano le cifre decimali del primo numero e (invertite) del secondo numero.

Esercizio 3: (PUNTI 8)
Dato il programma Prolog:

```
fits(X,X).
fits(node(L,R),X) :- fits(L,X).
fits(node(L,R),X) :- fits(R,X).

search(X,N,N):-atomic(X),!.
search(node(L,R),N_in,N_out) :-
    N1 is N_in+1,
    search(L,N1,N_temp),
    search(R,N_temp,N_out).
```

si rappresenti l'albero SLD relativo alla query:

```
:- search(node(node(1,2),3),0,X),fits(node(node(1,2),3),X).
```

Esercizio 4: (PUNTI 10)
Si scriva un meta-programma Prolog che prende in ingresso un programma P e un predicato G. Il programma deve fornire una lista di liste LL. Ogni elemento di LL contiene gli atomi del body di una clausola che unifica con G. Si suppone che il funtore di ogni atomo sia costituito da un solo carattere. I letterali del body devono essere ordinati in base all'ordine lessicografico dei loro funtori. Ad esempio, dato il programma:

```
f(D,S):- d(D,a,T),s(S,D),a(D,D),b(T,D).
f(3,S):- s(S,3),r(S),a(X,S).
```

il goal

```
:- meta(f(F,3),LL)
```

fornisce:

```
LL=[[a(F,F),b(T,F),d(F,a,T),s(3,F)],[a(X,3),r(3),s(3,3)]]
```

Si supponga di avere a disposizione un predicato del tipo `sort(L1,L2,L3)` che restituisce in L3 il contenuto di L1 ordinato secondo i valori crescenti contenuti in L2.
Esempio: `sort([a,b,c],[d,h,a],L3)`
fornisce:

```
L3 = [c,a,b]
```

SOLUZIONE

ESERCIZIO 1:

```
1: bambino(andrea)
2: ∀X ∀Y bambino(X) -> attivita(X,Y,scrive) ex-or attivita(X,Y,balla)
3: ∀X ∀Y bambino(X),attivita(X,Y,ascolta_musica) -> attivita(X,Y,balla)
4: attivita(andrea,mattino,ascolta_musica)
G: esiste X attivita(andrea,mattino,X)
```

Forma a clausole:

```
c1: bambino(andrea)
c2: not bambino(X) or attivita(X,Y,scrive) or attivita(X,Y,balla)
c3: not bambino(X) or not attivita(X,Y,scrive) or not attivita(X,Y,balla)
c4: not  bambino(X) or not attivita(X,Y,ascolta_musica) or
    attivita(X,Y,balla)
c5: attivita(andrea,mattino,ascolta_musica)
notG: not attivita(andrea,mattino,X)
```

Risoluzione:
Prima soluzione:

```
notG + c5 contraddizione X/ascolta_musica
```

Seconda soluzione:

```
c6=c1+c4: not attivita(andrea,Y,ascolta_musica) or
    attivita(andrea,Y,balla)
c7=c5+c6: attivita(andrea,Y,balla)
c8=c7+notG: clausola vuota
```

ESERCIZIO 2:

Variabili: A0, A1
Domini: A0, A1 :: [0..9]
Vincoli:

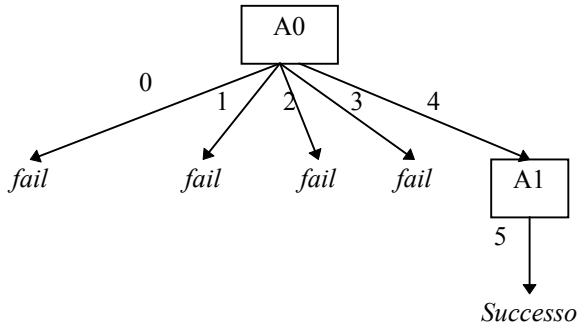
Invertendo le cifre di N2 si ottiene N1:	
N1 ha come cifre A0 ed A1	N1 = A1*10+A0
N2 ha come cifre A1 e A0	N2 = A0*10+A1
La somma delle cifre di N1 sommata ad N1 è N2:	A1*10+A0+A1+A0 = A0*10+A1
N1 è compreso tra 1 e 99	*1* <= A1*10+A0<=99

E' facile mostrare che la seguente rappresentazione è equivalente:

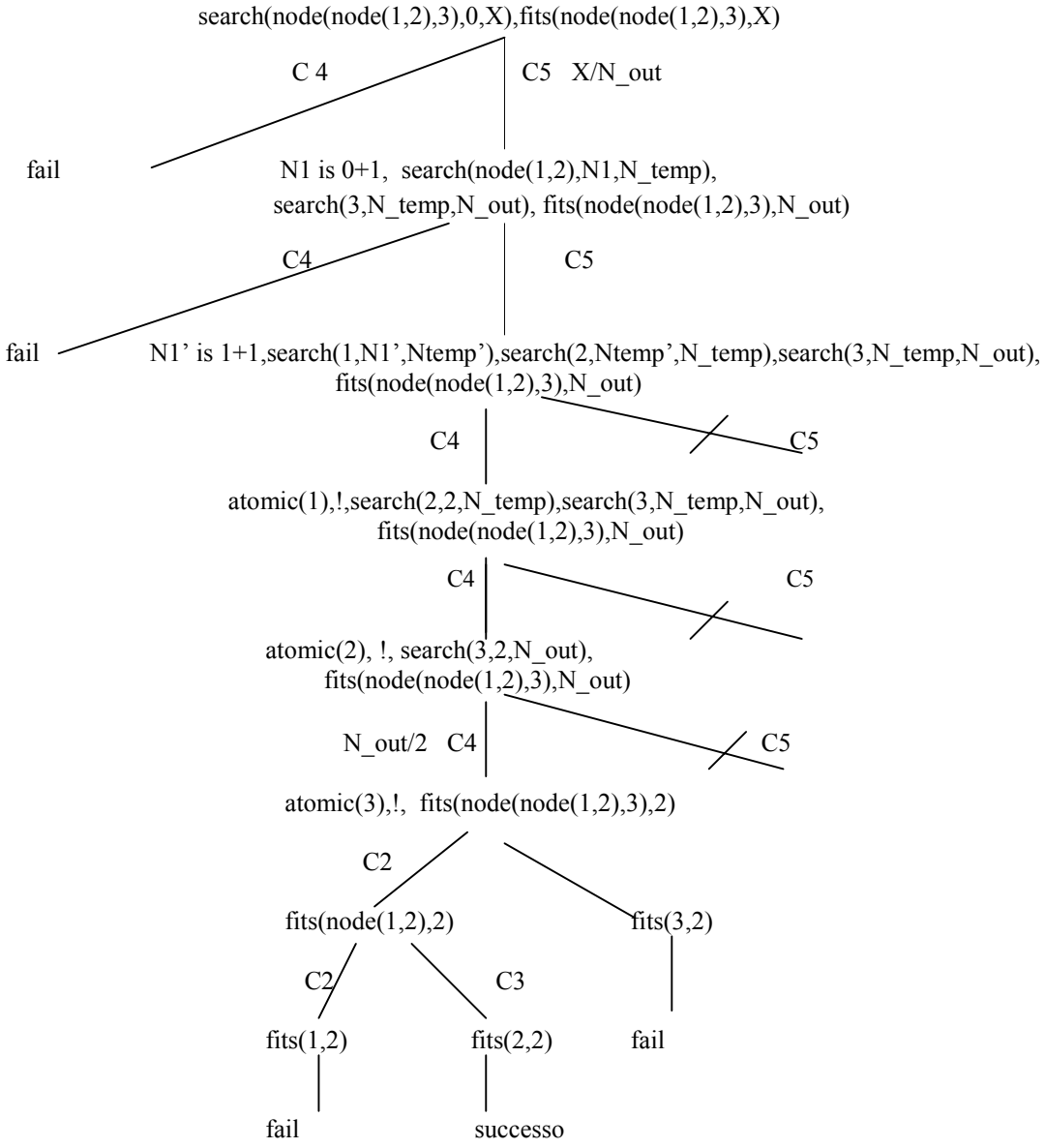
$1 \leq A0 + 10*A1 \leq 99$
$10*A1 = 8*A0$ ovvero $5*A1=4*A0$

Risoluzione

Utilizziamo il Forward Checking:


A0
0
1
2
3
4
fail
fail
fail
fail
A1
5
Successo

ESERCIZIO 3:


search(node(node(1,2),3),0,X),fits(node(node(1,2),3),X)
C 4
C5 X/N_out
fail
N1 is 0+1, search(node(1,2),N1,N_temp),
search(3,N_temp,N_out), fits(node(node(1,2),3),N_out)
C4
C5
fail
N1' is 1+1,search(1,N1',Ntemp'),search(2,Ntemp',N_temp),search(3,N_temp,N_out),
fits(node(node(1,2),3),N_out)
C4
C5
atomic(1),!,search(2,2,N_temp),search(3,N_temp,N_out),
fits(node(node(1,2),3),N_out)
C4
C5
atomic(2), !, search(3,2,N_out),
fits(node(node(1,2),3),N_out)
N_out/2 C4
C5
atomic(3),!, fits(node(node(1,2),3),2)
C2
fits(node(1,2),2)
fits(3,2)
C2
C3
fits(1,2)
fits(2,2)
fail
fail
successo

ESERCIZIO 4:

```
meta(G,LL) :-
    findall(Body,clause(G,Body),ListaBody),
    ordina_liste(ListaBody,LL).

ordina_liste([],[]).
ordina_liste([Body|Tail],[Sorted|NewTail]) :-
    converti(Body,Lista),
    ordina(Lista,Sorted),
    ordina_liste(Tail,NewTail).

% Converte il body in una lista di atomi
converti(true,[]) :- !.
converti((A,B),[A|NB]) :- !, converti(B,NB).
converti(X,[X]).

% Ordina gli atomi in base al funtore
ordina(L,Sorted) :-
    lista_funtori(L,LF),
    sort(L,LF,Sorted).

% Trova la lista dei funtori
lista_funtori([],[]).
lista_funtori([H|T],[F|TF]) :-
    functor(H,F,_),
     lista_funtori(T,TF).
```