COMPITO DI FONDAMENTI DI INTELLIGENZA ARTIFICIALE

1 Aprile 2003

Laurea Specialistica e I parte Vecchio Ordinamento (2 ore)

Esercizio 1 (punti 7)

Si traducano in logica del primo ordine le seguenti frasi di linguaggio comune:

1. I lupi mangiano gli agnelli, se questi aprono la porta di casa
2. Gli agnellini aprono la porta di casa solo a chi ha le zampe bianche quando sono in casa da soli
3. I lupi stupidi hanno le zampe nere
4. C'è un lupo furbo che si è colorato le zampe di bianco
5. Esiste un agnellino che è in casa da solo

Si utilizzi il principio di risoluzione per dimostrare che c'è un agnellino che viene mangiato da un lupo

Esercizio 2 (punti 8)

Si scriva un programma Prolog `list_abs(LISTIN,LISTOUT)`che data una lista di elementi `LISTIN`, produca una lista in uscita `LISTOUT` contenente come elementi i valori –1, 0, 1 a seconda che il corrispondente elemento in `LISTIN` sia negativo, positivo o nullo.

Esempio:

```
?- list_abs([-7,2,0,3,5],X).
```

restituisce:

```
X= [-1,1,0,1,1]
```

Esercizio 3 (punti 9)

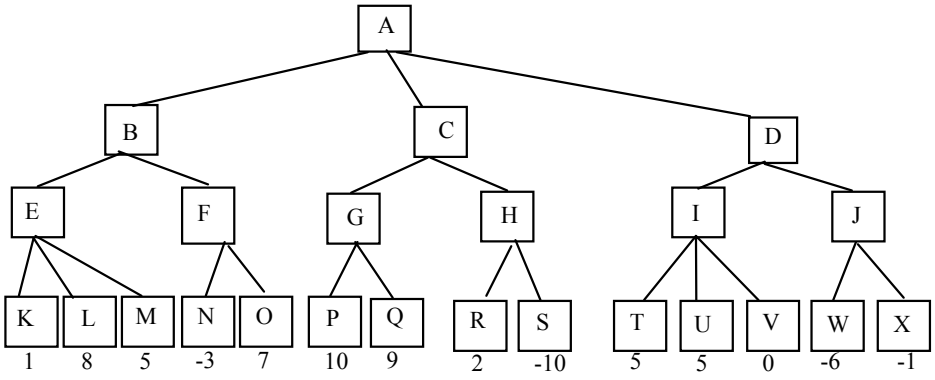
Si consideri il seguente problema: si devono allocare 6 task t1, …, t6 le cui durate sono rispettivamente 3,4,5,4,7,6 su due risorse m1 e m2.

- Ad ogni task è associata la risorsa su cui viene eseguito: (t1, m2), (t2,m1), (t3,m2), (t4,m1), (t5,m1) (t6,m2).
- Su ogni risorsa può essere in esecuzione un solo task alla volta.
- Esistono dei vincoli di precedenza tra le seguenti coppie di task (t2,t4), (t1,t3), (t4,t5), (t1,t6), in cui il secondo task deve iniziare dopo o nello stesso istante in cui il primo finisce su una qualunque risorsa.
- Le risorse sono disponibili dall'istante 1. E' infine necessario che tutti i task terminino entro l'istante di tempo 16,

Si modelli il problema come un problema di soddisfacimento di vincoli.
Sul problema iniziale, si applichi la arc consistenza ai soli vincoli di precedenza.
Infine lo si risolva utilizzando il forward checking.

Esercizio 4 (punti 4)

Si consideri il seguente albero di gioco dove i punteggi sono tutti dal punto di vista del primo giocatore:



Supponendo che il primo giocatore sia MIN, quale mossa dovrebbe scegliere? Si risolva l'esercizio tramite l'algoritmo MIN-MAX. Successivamente si mostrino i tagli alfa-beta.

Esercizio 5 (punti 4)

Il linguaggio Prolog non è un buon risolutore automatico della logica Classica

Si commenti questa frase esprimendo sinteticamente le ragioni di tale affermazione.

SOLUZIONE

Esercizio 1

Traduzione in clausole

I lupi mangiano gli agnelli, se questi aprono la porta di casa

$\forall$ L $\forall$ A lupo(L) $\wedge$ agnello(A) $\wedge$ apre(A) $\Rightarrow$ mangia(L,A).

1. ~ lupo(L) $\vee$ ~ agnello(A) $\vee$ ~ apre(A) $\vee$ mangia(L,A).

Gli agnellini aprono la porta di casa solo a chi ha le zampe bianche quando sono in casa da soli

$\forall$ A $\forall$ X agnello(A) $\wedge$ solo(A) $\wedge$ zampe(X,bianche) $\Rightarrow$ apre(A)

2. ~ agnello(A) $\vee$ ~ solo(A) $\vee$ ~ zampe(X,bianche) $\vee$ apre(A)

I lupi stupidi hanno le zampe nere

$\forall$ L lupo(L) $\wedge$ ~ furbo(L) $\Rightarrow$ zampe(L,nere)

3. ~ lupo(L) $\vee$ furbo(L) $\vee$ zampe(L,nere)

C'è un lupo furbo che si è colorato le zampe di bianco

$\exists$ L lupo(L) $\wedge$ furbo(L) $\wedge$ zampe(L, bianche)

4a.lupo(alberto)
4b. furbo(alberto)
4c. zampe(alberto, bianche)

Esiste un agnellino che è in casa da solo

$\exists$ A agnello(A) $\wedge$ solo(A).

5a. agnello(fiocco)
5b. solo(fiocco)

Goal: c'è un agnellino che viene mangiato da un lupo
$\exists$ A $\exists$ L, agnello(A) $\wedge$ lupo(L) $\wedge$ mangia(L,A)

Goal negato:
GN: ~ agnello(A) $\vee$ ~ lupo(L) $\vee$ ~ mangia(L,A)

Risoluzione
6 = GN+1: ~ lupo(L) $\vee$ ~ agnello(A) $\vee$ ~ apre(A)

7 = 6+2: $\sim$ lupo(L) $\vee \sim$ agnello(A) $\vee \sim$ solo(A) $\vee \sim$ zampe(X,bianche)
8 = 7+4c: $\sim$ lupo(alberto) $\vee \sim$ agnello(A) $\vee \sim$ solo(A)
9 = 8+4a: $\sim$ agnello(A) $\vee \sim$ solo(A)
10 = 9+5a: $\sim$ solo(fiocco)
11 = 10+5b: []

Esercizio 2

```
list_abs([], []).
list_abs([EL|Tail],[Val|B]):- abstract(A,Val), list_abs(Tail,B).

abstract(0,0):-!.
abstract (X,1):-X>0,!.
abstract (X,-1):- X<0.
```

Esercizio 3

Ad ogni task è associata una variabile che rappresenta l'inizio del task. Inoltre, ad ogni task è associata una variabile risorsa

Start1::[1..13] **R1=m2**
Start2::[1..12] **R2=m1**

Start3::[1..11] **R3=m2**
Start4::[1..12] **R4=m1**
Start5::[1..9] **R5=m1**
Start6::[1..10] **R6=m2**

Start2 + 4 ≤ Start4
Start1 + 3 ≤ Start3
Start4 + 4 ≤ Start5
Start1 + 3 ≤ Start6

Se **Ri = Rj** $\rightarrow$ **Starti + di ≤ Startj ∨ Startj + dj ≤ Starti**

Arc consistenza su vincoli di precedenza:

Start2 + 4 ≤ Start4 $\rightarrow$ **Start2::[1..8] Start4::[5..12]**
Start4 + 4 ≤ Start5 $\rightarrow$ **Start4::[5] Start5::[9]**

Risveglio
Start2 + 4 ≤ Start4 $\rightarrow$ **Start2::[1] Start4::[5]**

Start1 + 3 ≤ Start3 $\rightarrow$ **Start1::[1..8] Start3::[4..11]**
Start1 + 3 ≤ Start6 $\rightarrow$ **Start1::[1..7] Start6::[4..10]**

Arc consistenza su vincoli di precedenza:

Start1::[1..8]
Start2::[1]

Start3::[4..11]
Start4::[5]
Start5::[9]
Start6::[4..10]

Forward checking

Start2 = 1
Domini non cambiano

Start4 = 5
Domini non cambiano

Start5= 9
Domini non cambiano

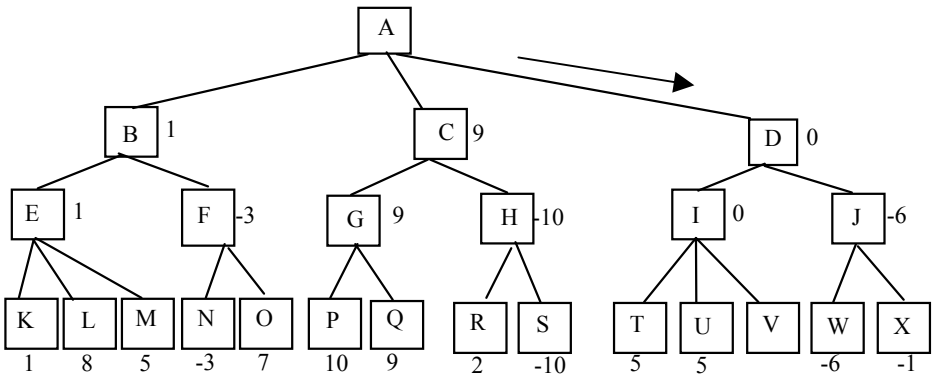
Start1 = 1
Domini non cambiano

Start3 = 4
Start6::[9..10]

Start6= 9

Esercizio 4

Min max



Alfa Beta