25 luglio 2017 - laboratorio

Constraint Programming

Prof. Marco Gavanelli

25 luglio 2017

Descrizione problema

Due genitori devono organizzare il mese di giugno per il loro figlio infante, e decidere con chi starà il bimbo ogni giorno del mese.

Hanno la possibilità di mandarlo ad un asilo estivo. L'asilo va prenotato, lo si può prenotare per un periodo costituito al massimo da 7 giorni; i giorni devono essere consecutivi. Al momento della prenotazione, si paga il costo per tutti i 7 giorni; il costo dell'asilo è di 100€. (Chiaramente, possono anche pagare i 100€ e non mandare il bambino all'asilo estivo). Ad esempio, se i genitori prenotano l'asilo per il periodo dal 4 al 10 giugno, poi possono mandare il bambino in quei giorni all'asilo. Potrebbero anche decidere di mandarlo per solo alcuni giorni in quell'intervallo; ad esempio il 7 e il 10; non potranno mandarlo all'asilo fuori da quell'intervallo. Come altro esempio, non è possibile mandare il bimbo il 5 e il 28 giugno, perché l'asilo può essere prenotato solo per un periodo di 7 giorni consecutivi.

Ciascuno dei genitori può prendersi dei giorni di ferie per stare con il bambino; il padre può prendersi fino a 4 giorni, mentre la madre può prendersi fino a 6 giorni. I genitori però non possono prendere le ferie in giorni in cui hanno appuntamenti o impegni lavorativi importanti. Gli impegli lavorativi dei genitori sono riportati in un file `impegni.pl`; sono indicati con dei fatti

```
impegno(Genitore,Giorno)
```

dove

- Genitore può essere 1 (per rappresentare il padre) o 2 (per rappresentare la madre)
- Giorno è un numero da 1 a 30 (i giorni del mese di giugno).

Infine, i genitori possono decidere di lasciare il bambino con una baby-sitter, al costo di 50€ al giorno.

Si trovi un'organizzazione in cui il bambino è ogni giorno con qualcuno e che minimizza il costo per i genitori.

CLP (15 punti)

Si risolva il problema usando ECL^{i}PS^e.

ASP (10 punti)

Si risolva il problema in Aswer Set Programming.

Soluzione

CLP

Variabili decisionali:

- Per ogni giorno del mese, una variabile della lista L dice con chi sta il bambino (con valori da 1 a 4 a seconda delle 4 opzioni)
- VaAsilo: è un boolean che dice se i genitori decidono di iscrivere o no il bambino all'asilo
- PrimoGiornoAsilo, UltimoGiornoAsilo: sono due variabili che rappresentano il primo e l'ultimo giorno in cui il bambino può andare all'asilo. Il loro dominio comprende tutti i giorni a disposizione nel mese

```
:- lib(fd).
:- lib(fd_global).
:- [impegni].

% Valori possibili
% con genitore X -> X (X=1..2)
% asilo estivo -> 3
% tata -> 4
estate(L):-
 length(L,30),
 L :: 1..4,
 impegni_genitori(L,1),
 occurrences(1,L,GiorniPadre),
 GiorniPadre :: 0..4,
 occurrences(2,L,GiorniMadre),
 GiorniMadre :: 0.. 6,
 occurrences(4,L,GiorniTata),
 Costo #= 10*IscrittoAsilo+5*GiorniTata,
 [PrimoGiornoAsilo,UltimoGiornoAsilo]::1..30,
 IscrittoAsilo::0..1,
 asilo(L,1,PrimoGiornoAsilo,UltimoGiornoAsilo,IscrittoAsilo),
 UltimoGiornoAsilo-PrimoGiornoAsilo #< 7,
 append(L,[PrimoGiornoAsilo,UltimoGiornoAsilo],Variabili),
 min_max((labeling(Variabili), myprint_list(L)),Costo).

asilo([],_,_,_,_).
asilo([H|T],N,Primo,Ultimo,IscrittoAsilo):-
 H #= 3 #<=> VaAsiloGiornoN,
% Se va all'asilo il giorno N, vuol dire che deve essere iscritto all'asilo
 VaAsiloGiornoN #=< IscrittoAsilo,
```

```
 Primo #=< N #<=> InizioAsiloPrimaDelGiornoN,
 Ultimo #>= N #<=> TerminaAsiloDopoIlGiornoN,
% Se nel giorno N il bimbo va all'asilo, vuol dire che il primo giorno di asilo
deve essere =< N
 VaAsiloGiornoN #=< InizioAsiloPrimaDelGiornoN,
% Se nel giorno N il bimbo va all'asilo, vuol dire che l'ultimo giorno di asilo
deve essere >= N
 VaAsiloGiornoN #=< TerminaAsiloDopoIlGiornoN,
 N1 is N+1,
 asilo(T,N1,Primo,Ultimo,IscrittoAsilo).

impegni_genitori([],_).
impegni_genitori([H|T],Giorno):-
 findall(Genitore,impegno(Genitore,Giorno),GenitoriImpegnati),
 elimina_valori(H,GenitoriImpegnati),
 Giorno1 is Giorno+1,
 impegni_genitori(T,Giorno1).

elimina_valori(_,[]).
elimina_valori(X,[H|T]):-
 X #\= H,
 elimina_valori(X,T).

% Stampa una lista di valori.
% Ovviamente non e` indispensabile, e` solo per avere un output piu` leggibile
myprint_list([]):-nl.
myprint_list([H|T]):- write(H), write(','), myprint_list(T).
```

ASP

```
giorno(1..30).
opzione(1..4). % 1=padre, 2=madre, 3=asilo, 4=tata

1{bambino(X,O):opzione(O)}1:- giorno(X).

:- impegno(Genitore,Giorno), bambino(Giorno,Genitore).

:- #count{Giorno:bambino(Giorno,1)} > 4.
:- #count{Giorno:bambino(Giorno,2)} > 6.

:- bambino(G1,3), bambino(G2,3), G2 - G1 >=7.

pagare_asilo :- bambino(G,3). % l'asilo va pagato se c'e` almeno un giorno in
cui ci va
```

```
costo_tata(C):- #count{G:bambino(G,4)} = GiorniTata, C=5*GiorniTata.

costo_tot(C):- costo_tata(C), not pagare_asilo.
costo_tot(Ctot):- costo_tata(Ctata), pagare_asilo, Ctot=Ctata+10.

#minimize {C:costo_tot(C)}.

#show bambino/2.
#show pagare_asilo.
```