

**Esercizio 1.** Svolgere tutti i punti.

a-1) Si consideri il seguente programma logico e se ne calcolino gli answer set, illustrando adeguatamente il procedimento seguito.

```
sbarco(Y) | nonSbarco(Y) :- not porto(Y), nave(Y,Z). 1  
porto(Y) :- nave(Y,X), not sbarco(Y). 2  
nave(X,Z) :- porto(X), sbarco(Y), Z=X+Y. 3  
  
nave(1,2).  
nave(4,5).
```

a-2) Si aggiunga il seguente strong constraint al programma del punto precedente.

```
:- nave(4,Y), #max{Z:sbarco(Z)} < Y.
```

Come influisce sulle soluzione del programma? Perché? Motivare adeguatamente la risposta.

a-1)

~~sbarco(1) | nonSbarco(1) :- not porto(1), nave(1,2).~~  
~~sbarco(4) | nonSbarco(4) :- not porto(4), nave(4,5).~~

~~porto(1) :- nave(1,2), not sbarco(1).~~  
~~porto(4) :- nave(4,5), not sbarco(4).~~

~~nave(1,2) :- porto(1), sbarco(1), Z=1+1.~~ % Non si potrà mai verificare  
~~nave(1,5) :- porto(1), sbarco(4), 5=1+4.~~  
~~nave(4,5) :- porto(4), sbarco(1), 5=4+1.~~ % Sempre sottopienette perché le altre vere  
~~nave(4,8) :- porto(4), sbarco(4), 8=4+4.~~ % come nave(1,2)

AS: {

A1: { [... EDB...], sbarco(1), sbarco(4) }

A2: { [... EDB...], sbarco(1), porto(4) }

A3: { [... EDB...], sbarco(4), porto(1), nave(1,5) }

A4: { [... EDB...], porto(1), porto(4) }

}

a-2)

$\vdash \text{have}(4, Y), \# \max \{ z : sbarco(z) \} < Y.$

Imperando questo strong constraint, renderebbero scartabili tutti gli AS, rendendo il programma inconsistente, "sbareo" poiché il valore più alto che può raggiungere è AL PIÙ 4, quindi  $< 5$ .

b) Si consideri ora un programma P (non è necessario sapere come è fatto) i cui answer set sono già stati calcolati e sono riportati di seguito.

A1: {s(1,2), b(2,2), h(1), a(1,2), h(2), a(2,2)}

A2: {s(1,2), b(2,2), h(1), a(1,2), b(2,1)}

A3: {s(1,2), b(2,2), h(1), k(1,2)}

Si supponga di aggiungere i seguenti weak constraint al programma P. Si calcoli quale sarebbe il costo di ognuno degli answer set riportati sopra, e si indichi quello ottimo, commentando il procedimento seguito.

% DLV syntax  
 $: \sim h(X), \# \sum \{ Y : a(X, Y) \} > 1. [X@1]$   
 $: \sim b(X, Y), k(\_, X). [1@X]$

% ASP-Core-2 syntax  
 $: \sim h(X), \# \sum \{ Y : a(X, Y) \} > 1. [X@1]$   
 $: \sim b(X, Y), k(\_, X). [1@X, X, Y]$

a-2)

A1: 3@1

#sum

$\sim$

$: \sim h(1), 2 > 1.$   
 $: \sim h(2), 2 > 1.$

[1@1]  
[2@1]

A2: 1@1 OPTIMUM

$: \sim h(1), 2 > 1.$

[1@1]

A3: 1@2

$: \sim b(1,2), k(1,2).$

[1@2, 2, 2]

L'AS ottimo è A2

**Esercizio 2.** Sia dato un grafo non orientato  $G = \langle V, E \rangle$ , tale che ogni arco  $e \in E$  sia etichettato con un intero positivo strettamente maggiore di zero, corrispondente ad un peso; sia dato poi un intero  $P_{MAX}$ , strettamente maggiore di zero. Si vuole selezionare un insieme di nodi  $vSEL \subseteq V$  tale che siano rispettate le condizioni riportate di seguito.

- (s1) Se un nodo è incluso in  $vSEL$ , allora nessun altro nodo ad esso adiacente nel grafo  $G$  può essere incluso in  $vSEL$ ; cioè, se  $v \in vSEL$ , non può esistere  $u \in vSEL$  t.c.  $(u, v) \in E$ .
  - (s2) Non è possibile che ci siano archi in  $V$  che non sono coperti da  $vSEL$ . Un arco si considera “coperto” da  $vSEL$  se almeno uno dei suoi estremi è incluso in  $vSEL$ ; cioè,  $(u, v) \in E$  è “coperto” se  $u \in vSEL$  oppure  $v \in vSEL$ .
  - (s3) Ogni nodo incluso in  $vSEL$  ha un costo, che è calcolato come il peso *minimo* degli archi su esso incidenti; inoltre, il costo di  $vSEL$  è determinato dalla somma del costo di ciascun nodo incluso in  $vSEL$ . Il costo di  $vSEL$  non può mai eccedere il valore  $P_{MAX}$  dato.
- Inoltre, si preferiscono soluzioni ottime in accordo alle preferenze espresse di seguito.
- (w1) *Cosa più importante*: la differenza tra  $P_{MAX}$  e il costo totale di  $vSEL$  deve essere la minima possibile.
  - (w2) *Cosa meno importante*: Identifichiamo con  $uMin$  (rispettivamente,  $uMax$ ) un nodo incluso in  $vSEL$  tale che il suo costo sia pari al minimo (rispettivamente, al massimo) assoluto (si noti che possono esistere più nodi con questa proprietà). Si preferisce evitare che una coppia di nodi del tipo  $\langle uMin, uMax \rangle$  non sia connessa (i.e., “raggiungibile”) in  $G$ .

Modello dei dati in INPUT:

- $\text{node}(V)$   $\leftarrow$  i nodi del grafo
- $\text{edge}(U, V, C)$   $\leftarrow$  gli archi del grafo, con i rispettivi costi (*relazione SIMMETRICA*)
- $\text{pMAX}(C)$   $\leftarrow$  il costo massimo

$\neg vSel(V) \mid vSel(V) :- node(V).$

% s1

$\vdash vSel(V_1), \text{edge}(V_1, V_2, -), vSel(V_2), V_1 \neq V_2.$

% s2

$\vdash \text{edge}(A, B, -), \neg vSel(A), \neg vSel(B).$

% s3

$\text{cost}(V, C) :- vSel(V), C = \#\min\{W, V, Z : \text{edge}(V, Z, W)\}.$

$\vdash \text{pMAX}(M), \text{Tot} = \#\sum\{C, V : \text{cost}(V, C)\}, \text{Tot} > M.$

% w1

$\vdash \text{pMAX}(M), \text{Tot} = \#\sum\{C, V : \text{cost}(V, C)\}, D = M - \text{Tot}. [D @ 2]$

% W2

reaches(A, B) :- edge(A, B).

reaches(A, C) :- reaches(A, B), edge(A, C).

minCost(C) :- C = # min{W, N : cost(N, W)}.

maxCost(C) :- C = # max{W, N : cost(N, W)}.

vMin(N) :- vSel(N), cost(N, C), minCost(C).

vMax(N) :- vSel(N), cost(N, C), maxCost(C).

:- vMin(N1), vMax(N2), reaches(N1, N2). [1@1, N1, N2]

