LOGICA PER LA PROGRAMMAZIONE - a.a. 2016-2017
Seconda prova di verifica intermedia - 22/12/2016 — Soluzioni Proposte

Attenzione: Le soluzioni che seguono sono considerate corrette dai docenti. Per ogni esercizio possono esistere altre soluzioni corrette, anche molto diverse da quelle proposte.

ESERCIZIO 1

Assumendo che P, Q, R e S contengano la variabile libera x, si provi che la seguente formula è valida:

$$\neg(\exists x \,.\, P \wedge (\neg S \Rightarrow \neg P)) \;\wedge\; (\forall x \,.\, Q \Rightarrow \neg R) \;\wedge\; (\exists x \,.\, \neg Q \Rightarrow P) \;\Rightarrow\; \neg(\forall x \,.\, R \wedge S)$$

SOLUZIONE ESERCIZIO 1

Semplifichiamo la conseguenza:

$$\begin{array}{ll}
 & \underline{\neg(\forall x \,.\, R \wedge S)} \\
\equiv & \{\text{ (De Morgan)}\} \\
 & (\exists x \,.\, \underline{\neg(R \wedge S))} \\
\equiv & \{\text{ (De Morgan)}\} \\
 & (\exists x \,.\, \neg R \vee \neg S)
\end{array}$$

A questo punto utilizzando la regola della **Skolemizzazione** è sufficiente dimostrare che:

$$\neg(\exists x \,.\, P \wedge (\neg S \Rightarrow \neg P)) \;\wedge\; (\forall x \,.\, Q \Rightarrow \neg R) \;\wedge\; (\exists x \,.\, \neg Q \Rightarrow P) \;\wedge\; (\neg Q(a) \Rightarrow P(a)) \;\Rightarrow\; (\exists x \,.\, \neg R \vee \neg S)$$

con a costante nuova. Per dimostrare la formula partiamo dalla premessa:

$$\begin{array}{ll}
 & \underline{\neg(\exists x \,.\, P \wedge (\neg S \Rightarrow \neg P)) \;\wedge\; (\forall x \,.\, Q \Rightarrow \neg R) \;\wedge\; (\exists x \,.\, \neg Q \Rightarrow P) \;\wedge\; (\neg Q(a) \Rightarrow P(a))} \\
\Rightarrow & \{(\text{sempl-}\wedge),\text{ occor. pos.}\} \\
 & \underline{\neg(\exists x \,.\, P \wedge (\neg S \Rightarrow \neg P))} \;\wedge\; (\forall x \,.\, Q \Rightarrow \neg R) \;\wedge\; (\neg Q(a) \Rightarrow P(a)) \\
\equiv & \{(\text{De Morgan})\} \\
 & \underline{(\forall x \,.\, \neg(P \wedge (\neg S \Rightarrow \neg P)))} \;\wedge\; \underline{(\forall x \,.\, Q \Rightarrow \neg R)} \;\wedge\; (\neg Q(a) \Rightarrow P(a)) \\
\Rightarrow & \{(\text{elim-}\forall),\text{ occor. pos.}\} \\
 & \underline{\neg(P(a) \wedge (\neg S(a) \Rightarrow \neg P(a)))} \;\wedge\; (Q(a) \Rightarrow \neg R(a)) \;\wedge\; (\neg Q(a) \Rightarrow P(a)) \\
\equiv & \{(\text{De Morgan})\} \\
 & (\neg P(a) \vee \neg(\underline{\neg S(a) \Rightarrow \neg P(a))}) \;\wedge\; \underline{(Q(a) \Rightarrow \neg R(a))} \;\wedge\; \underline{(\neg Q(a) \Rightarrow P(a))} \\
\equiv & \{(\text{elim-} \Rightarrow),(\text{Doppia Negazione})\} \\
 & (\neg P(a) \vee \neg(\underline{S(a) \vee \neg P(a))}) \;\wedge\; (\neg Q(a) \vee \neg R(a)) \;\wedge\; (Q(a) \vee P(a)) \\
\equiv & \{(\text{De Morgan})\} \\
 & (\underline{\neg P(a) \vee (\neg S(a) \;\wedge\; P(a))}) \;\wedge\; (\neg Q(a) \vee \neg R(a)) \;\wedge\; (Q(a) \vee P(a)) \\
\equiv & \{(\text{Complemento})\} \\
 & \underline{(\neg P(a) \vee \neg S(a)) \wedge} (\neg Q(a) \vee \neg R(a)) \;\wedge\; \underline{(Q(a) \vee P(a))} \\
\Rightarrow & \{(\text{Risoluzione}),\text{ occor. pos.}\} \\
 & \underline{(\neg S(a) \vee Q(a)) \wedge (\neg Q(a) \vee \neg R(a))} \\
\Rightarrow & \{(\text{Risoluzione}),\text{ occor. pos.}\} \\
 & \underline{\neg S(a) \vee \neg R(a)} \\
\Rightarrow & \{(\text{intro-}\exists),\text{ occor. pos. }\} \\
 & (\exists x \,.\, \neg R \vee \neg S)
\end{array}$$

ESERCIZIO 2

Assumendo **a, b: array [0, n) of int**, si formalizzi il seguente enunciato:

"Ogni elemento dell'array **a** è uguale alla somma dell'elemento corrispondente di **b** e della somma dei valori pari di **b**." **SOLUZIONE ESERCIZIO 2**

SOLUZIONE ESERCIZIO 2

$$(\forall i.\ i \in [0, n) \ \Rightarrow\ a[i] = b[i] + (\Sigma j : j \in [0, n) \ \wedge\ pari(b[j])\ .\ b[j]))$$

ESERCIZIO 3
Si dica se la seguente tripla è verificata. Se lo è, fornire una dimostrazione formale; se non lo è, fornire un controesempio.

$$\{x = A \ \wedge\ y = B\}\ \mathtt{z := y * x;\ y, z := y - x, y * z}\ \{z = A * B * (B - A)\}$$

SOLUZIONE ESERCIZIO 3
La tripla non è verificata. Per mostrarlo, forniamo un controesempio, cioè uno stato σ che

1. soddisfa la precondizione ($\sigma \models x = A \ \wedge\ y = B$), ma tale che
2. l'esecuzione del comando in σ porta in uno stato σ' che non soddisfa la postcondizione ($z = A * B * (B - A)$).

Consideriamo lo stato $\sigma = \{(x, 2), (y, 2), (z, 3)\}$. Eseguendo il primo assegnamento $\mathtt{z := y * x}$ nello stato σ otteniamo lo stato

$$\sigma_1 = \sigma[^4/_z] = \{(x, 2), (y, 2), (z, 4)\}.$$

Eseguendo il secondo assegnamento multiplo $\mathtt{y, z := y - x, y * z}$ nello stato σ_1 otteniamo lo stato

$$\sigma_2 = \sigma_1[^{0,8}/_{y,z}] = \{(x, 2), (y, 0), (z, 8)\}.$$

Si noti che lo stato σ_2 non soddisfa la postcondizione $z = A * B * (B - A)$. Infatti le variabili di specifica A e B si riferiscondo ai valori di x ed y nella precondizione. Quindi si dovrebbe avere $z = 2 * 2 * (2 - 2) = 0$.

ESERCIZIO 4
Assumendo **a, b: array [0, m) of int**, si verifichi la seguente tripla, in cui l'operatore binario `max` restituisce il maggiore tra i due operandi:

$$\{x \in [1, m) \ \wedge\ \big(\forall i\,.\, i \in [0, x) \ \Rightarrow\ b[i] = (\mathbf{max}\ j : j \in [0, i]\,.\, a[j])\big)\}$$

```
b[x] := b[x-1] max a[x]
```

$$\{\big(\forall i\,.\, i \in [0, x] \ \Rightarrow\ b[i] = (\mathbf{max}\ j : j \in [0, i]\,.\, a[j])\big)\}$$

SOLUZIONE ESERCIZIO 4

Applicando l'**Assioma dell'Aggiornamento Selettivo** e la regola (PRE), dobbiamo verificare che:

$$x \in [1, m) \ \wedge\ \big(\forall i\,.\, i \in [0, x) \ \Rightarrow\ b[i] = (\mathbf{max}\ j : j \in [0, i]\,.\, a[j])\big) \Rightarrow$$

$$x \in dom(b) \wedge def(x) \wedge def(b[x - 1] \ \mathrm{max}\ a[x]) \wedge \big(\forall i\,.\, i \in [0, x] \ \Rightarrow\ b[i] = (\mathbf{max}\ j : j \in [0, i]\,.\, a[j])\big)[^c/_b]$$

dove $c = b[^{b[x-1]\ \mathrm{max}\ a[x]}/_x]$.
Partiamo dalla conseguenza

$$\begin{array}{ll} & x \in dom(b) \wedge def(x) \wedge def(b[x - 1] \ \mathrm{max}\ a[x]) \wedge \big(\forall i\,.\, i \in [0, x] \ \Rightarrow\ b[i] = (\mathbf{max}\ j : j \in [0, i]\,.\, a[j])\big)[^c/_b] \\ \equiv & \{\text{definizione di } def\} \\ & x \in dom(b) \wedge x - 1 \in dom(b) \wedge \big(\forall i\,.\, i \in [0, x] \ \Rightarrow\ b[i] = (\mathbf{max}\ j : j \in [0, i]\,.\, a[j])\big)[^c/_b] \\ \equiv & \{\mathbf{Ip}\text{: } x \in [1, m) \wedge dom(b) = [0, m)\} \\ & \big(\forall i\,.\, i \in [0, x] \ \Rightarrow\ b[i] = (\mathbf{max}\ j : j \in [0, i]\,.\, a[j])\big)[^c/_b] \\ \equiv & \{\text{sostituzione}\} \end{array}$$

$\big(\forall i \,.\, i \in [0,x] \;\Rightarrow\; c[i] = (\mathbf{max}\ j \colon j \in [0,i] \,.\, a[j])\big)$

$\equiv$ $\{$(Intervallo-$\forall$), **Ip**: $x > 0\}$

$\big(\forall i \,.\, i \in [0,x) \;\Rightarrow\; c[i] = (\mathbf{max}\ j \colon j \in [0,i] \,.\, a[j])\big) \wedge c[x] = (\mathbf{max}\ j \colon j \in [0,x] \,.\, a[j])$

$\equiv$ $\big\{$definizione di $c = b[^{b[x-1] \text{ max } a[x]}/_x]\big\}$

$\big(\forall i \,.\, i \in [0,x) \;\Rightarrow\; b[i] = (\mathbf{max}\ j \colon j \in [0,i] \,.\, a[j])\big) \wedge (b[x-1] \text{ max } a[x]) = (\mathbf{max}\ j \colon j \in [0,x] \,.\, a[j])$

$\equiv$ $\big\{$**Ip**: $\big(\forall i \,.\, i \in [0,x) \;\Rightarrow\; b[i] = (\mathbf{max}\ j \colon j \in [0,i] \,.\, a[j])\big)\big\}$

$(b[x-1] \text{ max } a[x]) = (\mathbf{max}\ j \colon j \in [0,x] \,.\, a[j])$

$\equiv$ $\{$(Intervallo-max)$\}$

$(b[x-1] \text{ max } a[x]) = ((\mathbf{max}\ j \colon j \in [0,x) \,.\, a[j]) \text{ max } a[x])$

$\equiv$ $\{$**Ip**: $b[x-1] = (\mathbf{max}\ j \colon j \in [0,x) \,.\, a[j])\}$

T

ESERCIZIO 5

Assumendo **c: array [0, m) of int**, si consideri il seguente frammento di programma annotato:

$\{cond = \mathbf{true} \;\wedge\; z = 0 \;\wedge\; m \geq 1\}$

$\{\mathsf{Inv}\colon z \in [0,m] \;\wedge\; \big(cond \equiv (\forall x.x \in [0,z) \Rightarrow c[x] = c[0])\big)\}\{\text{t: } m - z\}$

```
while (z < m) do
     if (c[z] = c[0])
          then z := z + 1
          else cond,z:= false,m
     fi;


endw
```

$\{cond \equiv (\forall x.x \in [0,m) \Rightarrow c[x] = c[0])\}$

Si scrivano le ipotesi di progresso ed invarianza. Inoltre si dimostri l'ipotesi di invarianza.

SOLUZIONE ESERCIZIO 5

Invariante $Inv : z \in [0,m] \;\wedge\; \big(cond \equiv (\forall x.x \in [0,z) \Rightarrow c[x] = c[0])\big)$
Funzione di terminazione $t : m - z$

1. **Ipotesi di Invarianza**:

$$\{z \in [0,m] \;\wedge\; \big(cond \equiv (\forall x.x \in [0,z) \Rightarrow c[x] = c[0])\big) \wedge (z < m)\}$$

`if (c[z]=c[0])  then z:=z+1 else  cond,z:= false,m fi`

$$\{z \in [0,m] \;\wedge\; \big(cond \equiv (\forall x.x \in [0,z) \Rightarrow c[x] = c[0])\big) \;\wedge\; def(z < m)\}$$

2. **Ipotesi di Progresso**:

$$\{z \in [0,m] \;\wedge\; z \in [0,m] \;\wedge\; \big(cond \equiv (\forall x.x \in [0,z) \Rightarrow c[x] = c[0])\big) \wedge (z < m) \wedge m - z = V\}$$

`if (c[z]=c[0])  then z:=z+1 else  cond,z:= false,m fi`

$$\{m - z < V\}$$

Dimostriamo l'ipotesi di invarianza. Applicando la **Regola del Condizionale**, dobbiamo verificare che

(5.1.1) $Inv \wedge (z < m) \Rightarrow def(c[z] = c[0])$

(5.1.2) $\{Inv \wedge (z < m) \wedge (c[z] = c[0])\}$ `z:=z+1` $\{Inv \wedge def(z < m)\}$

(5.1.3) $\{Inv \wedge (z < m) \wedge \neg(c[z] = c[0])\}$ $\mathtt{cond, z} := \mathtt{false, m}$ $\{Inv \wedge def(z < m)\}$

(5.1.1) Abbiamo che

$$\begin{array}{ll} & def(c[z] = c[0]) \\ \equiv & \quad \{\text{definizione di } def\} \\ & 0 \in dom(c) \wedge z \in dom(c) \\ \equiv & \quad \{\mathbf{Ip}\text{: } dom(c) = [0, m), z \in [0, m], z < m\} \\ & \mathbf{T} \end{array}$$

(5.1.2) Per dimostrare la tripla applichiamo la **Regola dell'Assegnamento** e ci riduciamo a dimostrare

$$\begin{array}{l} Inv \wedge (z < m) \wedge (c[z] = c[0]) \;\Rightarrow \\ def(z+1) \wedge (z \in [0, m] \;\wedge\; \big(cond \equiv (\forall x.x \in [0, z) \Rightarrow c[x] = c[0])\big))[^{z+1}/_{z}] \end{array}$$

Partiamo dalla conseguenza, applicando la sostituzione

$$\begin{array}{ll} & def(z+1) \wedge z+1 \in [0, m] \;\wedge\; \big(cond \equiv (\forall x.x \in [0, z] \Rightarrow c[x] = c[0])\big) \\ \equiv & \quad \{\text{definizione di def}\} \\ & z+1 \in [0, m] \;\wedge\; \big(cond \equiv (\forall x.x \in [0, z] \Rightarrow c[x] = c[0])\big) \\ \equiv & \quad \{\mathbf{Ip}\text{: } (z \in [0, m]) \wedge (z < m)\} \\ & \big(cond \equiv (\forall x.x \in [0, z] \Rightarrow c[x] = c[0])\big) \\ \equiv & \quad \{(\text{Intervallo-}\forall)\} \\ & cond \equiv ((\forall x.x \in [0, z) \Rightarrow c[x] = c[0]) \wedge c[z] = c[0]) \\ \equiv & \quad \{\mathbf{Ip}\text{: } c[z] = c[0]\} \\ & cond \equiv ((\forall x.x \in [0, z) \Rightarrow c[x] = c[0]) \wedge \mathbf{T}) \\ \equiv & \quad \{(\text{unita})\} \\ & cond \equiv (\forall x.x \in [0, z) \Rightarrow c[x] = c[0]) \\ \equiv & \quad \big\{\mathbf{Ip}\text{: } cond \equiv (\forall x.x \in [0, z) \Rightarrow c[x] = c[0])\big)\big\} \\ & (\forall x.x \in [0, z) \Rightarrow c[x] = c[0]) \equiv (\forall x.x \in [0, z) \Rightarrow c[x] = c[0]) \\ & \mathbf{T} \end{array}$$

(5.1.3) Applicando la **Regola dell'Assegnamento Multiplo** ci riduciamo a dimostrare

$$\begin{array}{l} Inv \wedge (z < m) \wedge \neg(c[z] = c[0]) \;\Rightarrow \\ def(m) \wedge def(false) \wedge (z \in [0, m] \;\wedge\; \big(cond \equiv (\forall x.x \in [0, z) \Rightarrow c[x] = c[0])\big))[^{m, false}/_{z, cond}] \end{array}$$

Partiamo dalla conseguenza, applicando la sostituzione

$$\begin{array}{ll} & def(m) \wedge def(false) \wedge m \in [0, m] \;\wedge\; \big(\mathbf{F} \equiv (\forall x.x \in [0, m) \Rightarrow c[x] = c[0])\big) \\ \equiv & \quad \{\text{definizione di def}\} \\ & m \in [0, m] \;\wedge\; \big(\mathbf{F} \equiv (\forall x.x \in [0, m) \Rightarrow c[x] = c[0])\big) \\ \equiv & \quad \{\text{definizione di intervallo}\} \\ & \mathbf{F} \equiv (\forall x.x \in [0, m) \Rightarrow c[x] = c[0]) \\ \equiv & \quad \{(\text{Intervallo-}\forall),\ \mathbf{Ip}\text{: } z \in [0, m] \wedge z < m\} \\ & \mathbf{F} \equiv ((\forall x.x \in [0, m) \wedge x \neq z \Rightarrow c[x] = c[0]) \wedge c[z] = c[0]) \\ \equiv & \quad \{\mathbf{Ip}\text{: } \neg(c[z] = c[0])\} \\ & \mathbf{F} \equiv ((\forall x.x \in [0, m) \wedge x \neq z \Rightarrow c[x] = c[0]) \wedge \mathbf{F}) \\ \equiv & \quad \{(\text{zero})\} \\ & \mathbf{F} \equiv \mathbf{F} \\ & \mathbf{T} \end{array}$$