Scuola di Scienze Matematiche, Fisiche e Naturali
Corso di Laurea Magistrale in Informatica

Esercizi per il corso di

MODELLI DI SISTEMI SEQUENZIALI E CONCORRENTI

FRANCESCO MUCCI
francesco.mucci@stud.unifi.it
6173140

Docente: Prof. *Rosario Pugliese*

Anno Accademico 2021-2022
12 febbraio 2023

INDICE

1 PRELIMINARI MATEMATICI 1
Esercizio 2.7 . . . 1
Esercizio 2.14 . . . 4
Esercizio 3.16 . . . 5
Esercizio 4.6 . . . 6
2 SISTEMI SEQUENZIALI 8
Esercizio 5.5 . . . 8
Esercizio 6.6 . . . 9
Esercizio 6.16 . . . 11
Esercizio 7.6 . . . 13
Esercizio 8.5 . . . 15
Esercizio 9.9 . . . 16
3 SISTEMI CONCORRENTI 18
Esercizio 11.8 . . . 18
A APPENDICE: DIMOSTRAZIONI ADDIZIONALI 25
Bibliografia 27

1

PRELIMINARI MATEMATICI

ESERCIZIO 2.7

[1] Sia R una relazione binaria su A.

1. Si provi che la chiusura riflessiva di R coincide con la relazione

   $$R \cup Id_A$$

   dove $Id_A = \{(x, x) \mid x \in A\}$.

2. Si provi che la chiusura simmetrica di R coincide con la relazione

   $$R \cup R^{-1}.$$

3. Si provi che la chiusura transitiva di R coincide con la relazione

   $$\{(x, y) \mid \exists\, x_1, \ldots, x_n \text{ con } x_i \, R \, x_{i+1} \text{ per } 1 \leqslant i \leqslant n-1,\ x_1 = x \text{ e } x_n = y\}.$$

Teorema 2.7.1 (Chiusura riflessiva di una relazione binaria su un insieme A)**.** *Sia* R *una relazione binaria su* A *e* Id_A *la relazione identità su* A*. La chiusura riflessiva di* R *coincide con la relazione* $R \cup Id_A$*.*

Dimostrazione. Vogliamo quindi provare che $R \cup Id_A$ è la più piccola relazione riflessiva su A che contiene R; per far ciò andremo a mostrare che:

1. $R \subseteq R \cup Id_A$;
2. $\forall\, a \in A,\ (a, a) \in R \cup Id_A$;
3. $\forall\, S \subseteq A \times A,\ S$ riflessiva $\wedge\ R \subseteq S \implies (R \cup Id_A) \subseteq S$.

1. Per definizione di unione tra insiemi, R è ovviamente sottoinsieme di $R \cup Id_A$.
2. Per definizione di unione tra insiemi e di Id_A, risulta ovvio anche che $R \cup Id_A$ sia riflessiva.
3. Consideriamo una qualsiasi relazione $S \subseteq A \times A$ che sia riflessiva e che contenga R; essendo che S è riflessiva risulterà che $\forall\, a \in A,\ (a, a) \in S$, ma allora, per defizione di Id_A, abbiamo che $Id_A \subseteq S$. Per ipotesi sappiamo che $R \subseteq S$ ed abbiamo appena mostrato che $Id_A \subseteq S$, da questo, per definizione di unione tra insiemi, segue che $(R \cup Id_A) \subseteq S$.

□

Teorema 2.7.2 (Chiusura simmetrica di una relazione binaria su un insieme A)**.** *Sia* R *una relazione binaria su* A*. La chiusura simmetrica di* R *coincide con la relazione* $R \cup R^{-1}$*.*

Dimostrazione. Vogliamo quindi provare che $R \cup R^{-1}$ è la più piccola relazione simmetrica su A che contiene R; per far ciò andremo a mostrare che:

1. $R \subseteq R \cup R^{-1}$;
2. $\forall\, a, b \in A,\ (a, b) \in R \cup R^{-1} \implies (b, a) \in R \cup R^{-1}$;
3. $\forall\, S \subseteq A \times A,\ S$ simmetrica $\wedge\ R \subseteq S \implies (R \cup R^{-1}) \subseteq S$.

1. Per definizione di unione tra insiemi, R è ovviamente sottoinsieme di $R \cup R^{-1}$.
2. Per definizione di inversa di una relazione risulta che $\forall\, (a, b) \in R,\ (b, a) \in R^{-1}$, ma allora, per definizione di unione tra insiemi, abbiamo che $R \cup R^{-1}$ è sicuramente simmetrica.
3. Consideriamo una qualsiasi relazione $S \subseteq A \times A$ che sia simmetrica e che contenga R. Essendo che S è simmetrica risulterà che $\forall\, (a, b) \in S,\ (b, a) \in S$; di conseguenza, avendo assunto che $R \subseteq S$, sarà vero che $\forall\, (a, b) \in R,\ (b, a) \in S$. Possiamo dunque concludere che $R^{-1} \subseteq S$. Per ipotesi sappiamo che $R \subseteq S$ ed abbiamo appena mostrato che $R^{-1} \subseteq S$, da questo, per definizione di unione tra insiemi, segue che $(R \cup R^{-1}) \subseteq S$.

□

Teorema 2.7.3 (Chiusura transitiva di una relazione binaria su un insieme A)**.** *Sia* R *una relazione binaria su* A*. La chiusura transitiva di* R *coincide con la relazione* V *definita come segue* [1]

$$V \triangleq \{(a, b) \mid \exists\, x_1, \ldots, x_n \text{ con } x_i\, R\, x_{i+1} \text{ per } 1 \leqslant i \leqslant n-1,\ x_1 = a,\ x_n = b \text{ e } n \geqslant 2\}.$$

Dimostrazione. Vogliamo quindi provare che V è la più piccola relazione transitiva su A che contiene R; per far ciò andremo a mostrare che:

1. $R \subseteq V$;
2. $\forall\, a, b, c \in A,\ (a, b) \in V \wedge (b, c) \in V \implies (a, c) \in V$;
3. $\forall\, T \subseteq A \times A,\ T$ transitiva $\wedge\ R \subseteq T \implies V \subseteq T$.

1. Considerando la definizione di V, risulta che la seguente relazione sia un suo sottoinsieme:

   $$\{(a, b) \mid \exists\, x_1,\ x_2 \text{ con } x_1\, R\, x_2,\ x_1 = a \text{ e } x_n = b\}.$$

   Tale relazione coincide ovviamente con R e dunque quest'ultimo è contenuto in V.

1 Si potrebbe dimostrare che V coincide con R^{+}.

2. Consideriamo tre qualsiasi $a, b, c \in A$ tali che $(a, b) \in V \wedge (b, c) \in V$. Essendo che $(a, b) \in V$, per definizione di V, possiamo dire che

   $$\exists\, x_1, \ldots, x_j \text{ con } x_i \,R\, x_{i+1} \text{ per } 1 \leqslant i \leqslant j-1,\ x_1 = a,\ x_j = b \text{ e } j \geqslant 2.$$

   Essendo che $(b, c) \in V$, sempre per definizione di V, possiamo dire che

   $$\exists\, y_1, \ldots, y_k \text{ con } y_i \,R\, y_{i+1} \text{ per } 1 \leqslant i \leqslant k-1,\ y_1 = b,\ y_k = c \text{ e } k \geqslant 2.$$

   Rinominando gli y_i nel seguente modo $y_i = x_{j+i-1}$, cioè in modo tale che

   $$y_1 = x_j = b \text{ e } y_k = x_{j+k-1} = c,$$

   risulterà immediata la seguente conclusione:

   $$\exists\, x_1, \ldots, x_{j+k-1} \text{ con } x_i \,R\, x_{i+1} \text{ per } 1 \leqslant i \leqslant j+k-2,\ x_1 = a,\ x_{j+k-1} = c,$$
   $$\text{e con } j \geqslant 2 \text{ e } k \geqslant 2.$$

   Possiamo quindi concludere che $(a, c) \in V$ e, conseguentemente, affermare che V sia transitiva.

3. Consideriamo una qualsiasi relazione $T \subseteq A \times A$ che sia transitiva e che contenga R. Noi vogliamo mostrare che $V \subseteq T$ e cioè che

   $$\text{se } (a, b) \in V \text{ allora } (a, b) \in T.$$

   Riflettendo sulla definizione di V, vogliamo quindi dimostrare che $\forall\, n \geqslant 2$,

   $$\text{se } \exists\, x_1, \ldots, x_n :\ x_i \,R\, x_{i+1} \text{ per } 1 \leqslant i \leqslant n-1, \text{ con } x_1 = a \text{ e } x_n = b,$$
   $$\text{allora } (a, b) \in T.$$

   Procediamo nella dimostrazione lavorando per induzione su n.

   - *Caso base ($n = 2$).* Se $n = 2$ e $(x_1 = a, x_2 = b) \in R$, essendo che per ipotesi $R \subseteq T$, allora possiamo concludere che $(a, b) \in T$.
   - *Passo induttivo ($n = k + 1$).* Assumiamo che la tesi sia vera per $2 \leqslant n \leqslant k$ e che

     $$\exists\, (x_1 = a), \ldots, (x_{k+1} = b) :\ x_i \,R\, x_{i+1} \text{ per } 1 \leqslant i \leqslant k.$$

     Sappiamo dunque che:

     $$\exists\, (x_1 = a), \ldots, x_k :\ x_i \,R\, x_{i+1} \text{ per } 1 \leqslant i \leqslant k-1, \tag{3.1}$$
     $$(x_k, b) \in R \text{ con } x_k \,R\, b. \tag{3.2}$$

     Per il punto 3.1 e per l'ipotesi induttiva, possiamo affermare che $(a, x_k) \in T$; inoltre, per il punto 3.2 e dato che per ipotesi $R \subseteq T$, possiamo dedurre che $(x_k, b) \in T$. Possiamo infine concludere che $(a, b) \in T$ dato che, per ipotesi, T è transitiva ed abbiamo appena provato che $(a, x_k) \in T \wedge (x_k, b) \in T$.

□

ESERCIZIO 2.14

[1] Sia $\prec$ una relazione ben fondata su un insieme A. Si dimostri che:

1. la sua chiusura transitiva $\prec^+$ è ben fondata;
2. la sua chiusura riflessiva e transitiva $\prec^*$ è un ordinamento parziale.

Teorema 2.14.1 ($\prec^+$ è ben fondata)**.** *Sia A un insieme e sia $\prec \subseteq A \times A$ una relazione ben fondata, possiamo provare che anche $\prec^+$ è ben fondata.*

Dimostrazione. Per dimostrare che $\prec^+$ è ben fondata dobbiamo provare che ogni sottoinsieme non vuoto di A ha almeno un elmento minimale secondo $\prec^+$:

$$\forall\, S \subseteq A,\ S \neq \emptyset \implies \exists\, m \in S :\ \forall\, a \in A,\ a \prec^+ m \implies a \notin S.$$

Consideriamo un qualsiasi $S \subseteq A$ tale che $S \neq \emptyset$; essendo che $\prec^+$ è la chiusura transitiva di $\prec$, per il teorema 2.7.2 abbiamo che

$$\forall\, s, s' \in S :\ s \prec^+ s',\quad \exists\, x_1, \ldots, x_n :\ x_i \prec x_{i+1} \text{ per } 1 \leqslant i \leqslant n-1 \text{ con } x_1 = s,\ x_n = s' \text{ e } n \geqslant 2.$$

Possiamo di conseguenza costruire l'insieme S_{seq} degli elementi di tutte le possibili sequenze che "collegano" tramite $\prec$ due qualsiasi elementi di S:

$$S_{seq} \triangleq \bigcup_{s,s' \in S} \{x_1, \ldots, x_n \mid x_i \prec x_{i+1} \text{ per } 1 \leqslant i \leqslant n-1 \text{ con } x_1 = s,\ x_n = s' \text{ e } n \geqslant 2\}$$

Per costruzione di S_{seq} risulta che

$$S \neq \emptyset \implies S_{seq} \neq \emptyset.$$

Essendo che $\prec$ è ben fondata per ipotesi, possiamo affermare che

$$S_{seq} \neq \emptyset \implies \exists\, m_{seq} \in S_{seq} :\ m_{seq} \text{ è } \prec \text{ minimale di } S_{seq}.$$

Per costruzione di S_{seq}, sappiamo che m_{seq} appartiene ad una sequenza che collega due elementi di S:

$$m_{seq} \in S_{seq} \implies m_{seq} \in \{x_1, \ldots, x_n \mid x_i \prec x_{i+1} \text{ per } 1 \leqslant i \leqslant n-1 \text{ con } x_1 = s,\ x_n = s' \text{ e } n \geqslant 2\}.$$

Assumiamo per assurdo che m_{seq} non sia estremo sinistro della sequenza a cui appartiene: per quanto enunciato sopra, risulterà che

$$m_{seq} = x_k \text{ con } k \geqslant 2 \implies x_{k-1} \prec m_{seq},$$

ma questo non è possibile dato che m_{seq} è minimale di S_{seq} secondo $\prec$. Risulta dunque che m_{seq} è sempre estremo sinistro della sequenza di appartenenza.

Assumiamo per assurdo che m_{seq} non sia minimale di S secondo $\prec^+$:
per definizione di minimale, sappiamo che

$$m_{seq} \text{ non è } \prec^+ \text{ minimale di } S \implies \exists\, s \in S :\ s \prec^+ m_{seq};$$

per definizione di chiusura transitiva, possiamo inoltre affermare che

$$s \prec^+ m_{seq} \implies \exists\, x_1, \ldots, x_n :\ x_i \prec x_{i+1} \text{ per } 1 \leqslant i \leqslant n-1 \text{ con } x_1 = s,\ x_n = m_{seq} \text{ e } n \geqslant 2.$$

Abbiamo quindi individuato una sequenza che collega due elementi di S in cui m_{seq} non è l'estremo sinistro, ma questo non è possibile per quanto detto precedentemente. Possiamo quindi concludere che S ha un elemento minimale secondo $\prec^+$ e tale elemento è proprio m_{seq}. □

Teorema 2.14.2 ($\prec^*$ è un ordinamento parziale)**.** *Sia A un insieme e sia $\prec \subseteq A \times A$ una relazione ben fondata, possiamo provare che $\prec^*$ è un ordinamento parziale.*

Dimostrazione. Per provare che $\prec^*$ è una una relazione d'ordine parziale su A, dobbiamo mostrare che $\prec^*$ è riflessiva, transitiva e antisimmetrica.
Per definizione di chiusura riflessiva e transitiva, sappiamo che $\prec^*$ è ovviamente riflessiva e transitiva; dunque, ci rimane solo da dimostrare che è anche antisimmetrica:

$$\forall\, a, b \in A,\ a \prec^* b \wedge b \prec^* a \implies a = b.$$

Consideriamo due qualsiasi $a, b \in A$ tali che $a \prec^* b \wedge b \prec^* a$ e assumiamo per assurdo che $a \neq b$.
Essendo che $\prec^* = \mathrm{Id}_A \cup \prec^+$, risulta che

$$a \prec^* b \wedge a \neq b \implies a \prec^+ b;$$
$$b \prec^* a \wedge a \neq b \implies b \prec^+ a.$$

Dato che $\prec^+$ è transitiva, abbiamo che

$$a \prec^+ b \wedge b \prec^+ a \implies a \prec^+ a;$$

ma questo è impossibile essendo che, per il teorema 2.14.1, $\prec^+$ è ben fondata e una relazione ben fondata deve essere necessariamente irriflessiva:

$$\nexists\, a \in A :\ a \prec^+ a.$$

Deve quindi essere che $a = b$ e, di conseguenza, possiamo affermare che $\prec^*$ è, oltre che riflessiva e transitiva, anche antisimmerica: abbiamo quindi provato che $\prec^*$ è una relazione d'ordine parziale su A. □

ESERCIZIO 3.16

[1] Dimostrare che

$$\forall\, E, E_1, E_2 \in L(G_a),\ \forall\, n \in \mathbb{N} :\ E \to E_1,\ E \to E_2,\ E_1 \longrightarrow\!\!\!\!\!\rightarrow n \text{ implica } E_2 \longrightarrow\!\!\!\!\!\rightarrow n.$$

Teorema 3.16.1. *Sia G_a la grammatica che definisce la sintassi astratta delle espressioni aritmetiche e siano $\to$ e $\twoheadrightarrow$ le relazioni di transizione della semantica, rispettivamente, di computazione e di valutazione delle espressioni aritmetiche [1]. Possiamo provare che*

$$\forall\, E, E_1, E_2 \in L(G_a),\ \forall\, n \in \mathbb{N} :\ E \to E_1 \ \wedge\ E \to E_2 \ \wedge\ E_1 \twoheadrightarrow n \implies E_2 \twoheadrightarrow n.$$

Dimostrazione. Per l'equivalenza tra la semantica di computazione e la semantica di valutazione, sappiamo che

$$E_1 \twoheadrightarrow n \iff E_1 \xrightarrow{*} n,$$
$$E_1 \twoheadrightarrow n \implies \exists\, k \in \mathbb{N} :\ Op(E_1) = k \ \wedge\ E_1 \xrightarrow{k} n.$$

Di conseguenza, per la definizione di chiusura riflessiva e transitiva della relazione $\to$, abbiamo che

$$E \to E_1 \xrightarrow{k} n \equiv E \xrightarrow{*} n.$$

Inoltre, per la proposizione 3.17 delle note [1], risulta che

$$E \to E_1 \ \wedge\ Op(E_1) = k \implies Op(E) = 1 + k,$$
$$E \to E_2 \implies Op(E) = 1 + Op(E_2).$$

Sappiamo quindi che $Op(E_2) = k$.
A questo punto supponiamo per assurdo che $E_2 \xrightarrow{*} m$ con $m \neq n$. Per la proposizione 3.18 [1], possiamo affermare che

$$E_2 \xrightarrow{*} m \ \wedge\ Op(E_2) = k \iff E_2 \xrightarrow{k} m;$$

risulta quindi che $E \to E_2 \xrightarrow{k} m \equiv E \xrightarrow{*} m$ con $m \neq n$, ma questo non è possibile per via del determinismo della semantica di computazione:

$$E \xrightarrow{*} n \ \wedge\ E \xrightarrow{*} m \implies m = n.$$

Abbiamo dunque provato che $E_2 \xrightarrow{*} n$, ma allora, per l'equivalenza tra la semantica di computazione e quella di valutazione, deve essere che $E_2 \twoheadrightarrow n$. □

ESERCIZIO 4.6

[1] Costruire gli automi associati alle seguenti espressioni regolari:

1. $a; (b + c)$;
2. $a; b + a; c$.

Esercizio 4.6.1 (*Automa associato a* $a; (b + c)$)**.**
L'automa associato all'espressione regolare $a; (b + c)$ è mostrato figura 1; le transizioni

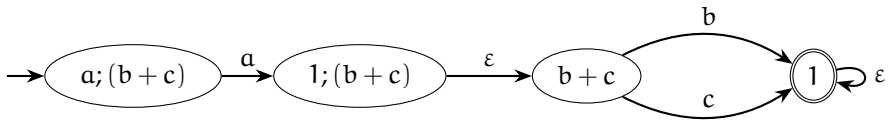

Figura 1: Automa associato a $a; (b + c)$.

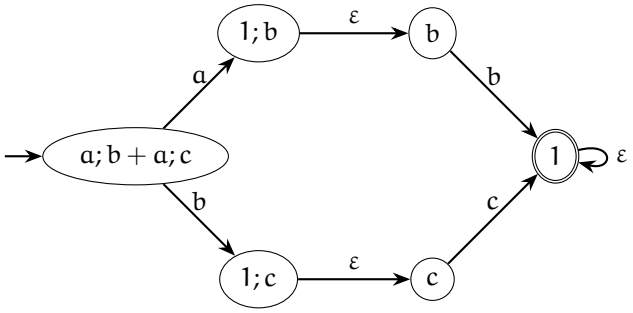

Figura 2: Automa associato a $a; b + a; c$.

necessarie per costruirlo sono state derivate nel seguente modo:

$$\dfrac{\dfrac{}{a \xrightarrow{a} 1}\ \textsc{Atom}}{a; (b + c) \xrightarrow{a} 1; (b + c)}\ \textsc{Seq1} \qquad \dfrac{\dfrac{}{1 \xrightarrow{\varepsilon} 1}\ \textsc{Tic}}{1; (b + c) \xrightarrow{\varepsilon} (b + c)}\ \textsc{Seq2}$$

$$\dfrac{\dfrac{}{b \xrightarrow{b} 1}\ \textsc{Atom}}{(b + c) \xrightarrow{b} 1}\ \textsc{Sum1} \qquad \dfrac{\dfrac{}{c \xrightarrow{c} 1}\ \textsc{Atom}}{(b + c) \xrightarrow{c} 1}\ \textsc{Sum2} \qquad \dfrac{}{1 \xrightarrow{\varepsilon} 1}\ \textsc{Tic}$$

Esercizio 4.6.2 (*Automa associato a* $a; b + a; c$)**.**
L'automa associato all'espressione regolare $a; b + a; c$ è visionabile in figura 2 e le sue transizioni sono state derivate nel seguente modo:

$$\dfrac{\dfrac{\dfrac{}{a \xrightarrow{a} 1}\ \textsc{Atom}}{a; b \xrightarrow{a} 1; b}\ \textsc{Seq1}}{a; b + a; c \xrightarrow{a} 1; b}\ \textsc{Sum1} \qquad \dfrac{\dfrac{\dfrac{}{a \xrightarrow{a} 1}\ \textsc{Atom}}{a; c \xrightarrow{a} 1; c}\ \textsc{Seq1}}{a; b + a; c \xrightarrow{a} 1; c}\ \textsc{Sum2}$$

$$\dfrac{\dfrac{}{1 \xrightarrow{\varepsilon} 1}\ \textsc{Tic}}{1; b \xrightarrow{\varepsilon} b}\ \textsc{Seq2} \qquad \dfrac{\dfrac{}{1 \xrightarrow{\varepsilon} 1}\ \textsc{Tic}}{1; c \xrightarrow{\varepsilon} c}\ \textsc{Seq2}$$

$$\dfrac{}{b \xrightarrow{b} 1}\ \textsc{Atom} \qquad \dfrac{}{c \xrightarrow{c} 1}\ \textsc{Atom} \qquad \dfrac{}{1 \xrightarrow{\varepsilon} 1}\ \textsc{Tic}$$

2

SISTEMI SEQUENZIALI

ESERCIZIO 5.5

[1] Dimostrare che **SK** = **KI** dove

$$\mathbf{S} \equiv \lambda xyz.xz(yz), \qquad \mathbf{K} \equiv \lambda xy.x, \qquad \mathbf{I} \equiv \lambda x.x.$$

Lemma 5.5.1. *Siano* $\mathbf{S} \equiv \lambda xyz.xz(yz)$ *e* $\mathbf{K} \equiv \lambda xy.x \equiv \lambda uv.u$*, possiamo provare che* **SK** *è* β*-riducibile in* $\lambda yz.z$*.*

Dimostrazione. β-riduciamo **SK** seguendo la strategia di riduzione normal-order:

$$\begin{aligned}\mathbf{SK} \;&\equiv\; (\lambda xyz.xz(yz))(\lambda uv.u)\\ &\longrightarrow_\beta \lambda yz.(\lambda uv.u)z(yz)\\ &\longrightarrow_\beta \lambda yz.(\lambda v.z)(yz)\\ &\longrightarrow_\beta \lambda yz.z.\end{aligned}$$

Abbiamo quindi mostrato che $\mathbf{SK} \Longrightarrow_\beta \lambda yz.z$ [1]. □

Lemma 5.5.2. *Siano* $\mathbf{K} \equiv \lambda xy.x \equiv \lambda uv.u$ *e* $\mathbf{I} \equiv \lambda x.x$*, possiamo provare che* **KI** *è* β*-riducibile in* $\lambda vx.x$*.*

Dimostrazione. β-riduciamo **KI** seguendo la strategia di riduzione normal-order:

$$\mathbf{SK} \equiv (\lambda uv.u)(\lambda x.x) \longrightarrow_\beta \lambda v.\lambda x.x \equiv \lambda vx.x.$$

Abbiamo quindi mostrato che $\mathbf{KI} \Longrightarrow_\beta \lambda vx.x$. □

Teorema 5.5.3. *Siano* **S**, **K** *e* **I** *i termini del lambda calcolo definiti nei lemmi 5.5.1 e 5.5.2, possiamo provare che* $\mathbf{SK} =_\beta \mathbf{KI}$*.*

Dimostrazione. Come conseguenza dei lemmi 5.5.1 e 5.5.2 e per la definizione di β-congruenza, possiamo affermare che:

$$\begin{aligned}&\mathbf{SK} \Longrightarrow_\beta \lambda yz.z \text{ implica } \mathbf{SK} =_\beta \lambda yz.z,\\ &\mathbf{KI} \Longrightarrow_\beta \lambda vx.x \text{ implica } \mathbf{KI} =_\beta \lambda vx.x.\end{aligned}$$

Dato che $\lambda yz.z \equiv \lambda vx.x$ e $=_\beta$ è una relazione di congruenza, possiamo concludere che $\mathbf{SK} =_\beta \mathbf{KI}$. □

1 $\lambda yz.z$ non contiene alcun β-redesso, risulta quindi essere una β-forma normale.

ESERCIZIO 6.6

[1] Si studi l'insieme V^* delle stringhe finite sull'alfabeto $V = \{a, b, c\}$, con l'ordinamento $\alpha \sqsubseteq \alpha\beta$, dove $\alpha\beta$ indica la concatenazione delle stringhe α e β.

1. La struttura $(V^*, \sqsubseteq)$ è un ordinamento parziale?
2. Esiste l'elemento minimo?
3. È completo?

Teorema 6.6.1 ($(V^*, \sqsubseteq)$ è un poset)**.** *Sia* $V = \{a, b, c\}$ *e sia* $\sqsubseteq$ *una relazione binaria su* V^* *tale che* $\alpha \sqsubseteq \beta$ *se e solo se* $\exists\, x \in V^* :\ \beta = \alpha x$*. La struttura* $(V^*, \sqsubseteq)$ *è un ordinamento parziale.*

Dimostrazione. Per provare che la struttura $(V^*, \sqsubseteq)$ è un ordinamento parziale ci basta mostrare che $\sqsubseteq$ è una relazione d'ordine parziale su V^* e cioè che $\sqsubseteq$ è

1. riflessiva: $\forall\, s \in V^*,\ s \sqsubseteq s$;
2. transitiva: $\forall\, \alpha, \beta, \gamma \in V^*,\ \alpha \sqsubseteq \beta \wedge \beta \sqsubseteq \gamma \implies \alpha \sqsubseteq \gamma$;
3. antisimmetrica: $\forall\, \alpha, \beta \in V^*,\ \alpha \sqsubseteq \beta \wedge \beta \sqsubseteq \alpha \implies \alpha = \beta$.

1. *Proviamo che* $\sqsubseteq$ *è riflessiva*. Essendo che la stringa vuota ε è l'elemento neutro dell'operazione di concatenazione tra stringhe, sappiamo che

   $$\forall\, s \in V^*,\ s = s\varepsilon;$$

   dunque, per definizione di $\sqsubseteq$, risulta che

   $$s = s\varepsilon \implies s \sqsubseteq s.$$

2. *Proviamo che* $\sqsubseteq$ *è transitiva*. Siano α e β, stringhe finite sull'alfabeto V, tali che $\alpha \sqsubseteq \beta \wedge \beta \sqsubseteq \gamma$. Per definizione di $\sqsubseteq$, abbiamo che

   $$\alpha \sqsubseteq \beta \implies \exists\, x \in V^* :\ \beta = \alpha x;$$
   $$\beta \sqsubseteq \gamma \implies \exists\, y \in V^* :\ \gamma = \beta y.$$

   Risulta quindi immediato che

   $$\beta = \alpha x \wedge \gamma = \beta y \implies \gamma = \alpha z \text{ con } z = xy.$$

   Infine, per definizione di $\sqsubseteq$, possiamo concludere che

   $$\gamma = \alpha z \implies \alpha \sqsubseteq \gamma.$$

3. *Proviamo che* $\sqsubseteq$ *è antisimmetrica*. Siano α e β, stringhe finite sull'alfabeto V, tali che $\alpha \sqsubseteq \beta \wedge \beta \sqsubseteq \alpha$. Per definizione di $\sqsubseteq$, abbiamo che

   $$\alpha \sqsubseteq \beta \implies \exists\, x \in V^* :\ \beta = \alpha x;$$
   $$\beta \sqsubseteq \alpha \implies \exists\, y \in V^* :\ \alpha = \beta y.$$

Risulta quindi immediato che

$$\beta = \alpha x \ \wedge \ \alpha = \beta y \implies \alpha = \alpha x y.$$

Essendo che la ε è l'elemento neutro della concatenazione, possiamo affermare che

$$\alpha = \alpha x y \implies x = \varepsilon \ \wedge \ y = \varepsilon.$$

Possiamo quindi concludere che

$$y = \varepsilon \ \wedge \ \alpha = \beta y \implies \alpha = \beta.$$

□

Teorema 6.6.2 (($V^*, \sqsubseteq$) ha elemento minimo)**.** *Sia* $(V^*, \sqsubseteq)$ *il poset definito nel teorema 6.6.1, questo ha come elemento minimo la stringa vuota* ε [2]*.*

Dimostrazione. Sappiamo che la stringa vuota è l'elemento neutro dell'operazione di concatenazione tra stringhe, possiamo dunque affermare che

$$\forall \ s \in V^*, \ s = \varepsilon s.$$

Per definizione di $\sqsubseteq$, risulta immediatamente che

$$s = \varepsilon s \implies \varepsilon \sqsubseteq s.$$

Abbiamo provato che $\forall \ s \in V^*, \ \varepsilon \sqsubseteq s$, possiamo infine concludere che ε è l'elemento minimo del poset $(V^*, \sqsubseteq)$. □

Teorema 6.6.3 (($V^*, \sqsubseteq$) non è un CPO)**.** *Sia* $(V^*, \sqsubseteq)$ *il poset definito nel teorema 6.6.1, questo non è completo.*

Affinché sia un CPO, $(V^*, \sqsubseteq)$ deve soddisfare le seguenti condizioni:

1. ha elemento minimo;
2. ogni catena in $(V^*, \sqsubseteq)$ ha estremo superiore.

Dimostrazione. Abbiamo già provato che l'elemento minimo di $(V^*, \sqsubseteq)$ è la stringa vuota ε, tuttavia possiamo provare che non tutte le catene in $(V^*, \sqsubseteq)$ hanno un estremo superiore. In modo informale possiamo giungere a questa conclusione riflettendo sul fatto che

- in $(V^*, \sqsubseteq)$ possiamo trovare delle catene numerabili [3];
- V^* contiene unicamente stringhe finite.

Se consideriamo una qualsiasi catena numerabile in $(V^*, \sqsubseteq)$, questa avrà un estremo superiore solo se da un certo punto in poi gli elementi della catena assumono un valore costante. Se ciò non succede, l'unico estremo superiore possibile sarebbe una stringa ω di lunghezza infinita che risulti elemento assorbente per la concatenazione di stringhe; tuttavia un tale estremo superiore non potrà esistere in quanto V^* contiene unicamente

2 Essendo che $(V^*, \sqsubseteq)$ è un poset, se questo ha un minimo, tale minimo è unico.

3 Con catena numerabile intendiamo un poset totalmente ordinato il cui insieme di riferimento ha cardinalità pari a $\aleph_0$.

stringhe finite su V.
Per provare formalmente quanto espresso a parole ci basta trovare un controesempio, cioè una catena in $(V^*, \sqsubseteq)$ per cui non esiste estremo superiore. Consideriamo l'insieme delle stringhe di lunghezza finita qualsiasi composte unicamente dal carattere a:

$$A^* = \{a\}^* = \{\varepsilon, a, aa, aaa, \ldots, a \ldots a, \ldots\}.$$

Lavorando per induzione sulla lunghezza delle stringhe di A^*, si può facilmente vedere che [4]

$$A^* \subseteq V^*;$$
$$\varepsilon \sqsubseteq a \sqsubseteq aa \sqsubseteq aaa \sqsubseteq \ldots \sqsubseteq a \ldots a \sqsubseteq \ldots.$$

Risulta dunque chiaro che $(A^*, \sqsubseteq)$ sia una catena in $(V^*, \sqsubseteq)$ e, in particolare, una catena numerabile dato che i suoi elementi possono essere messi in corrispondenza biunivoca con i naturali nel seguente modo:

$$a^0 = \varepsilon;$$
$$a^{n+1} = aa^n = a^n a \;\forall\, n \geqslant 0.$$

Se indichiamo la lunghezza di una stringa s con $|s|$, risulterà inoltre che

$$\forall\, a^n \in A^*,\ |a^n| = n.$$

Essendo che l'estremo superiore è il minimo dell'insieme dei maggioranti di A^* in V^*, per mostrare quanto desiderato ci basterà provare che tale insieme è vuoto. Ragioniamo per assurdo e assumiamo che esista un maggiorante di A^* in V^*:

$$\exists\, \alpha \in V^* :\ \forall\, a^n \in A^*,\ a^n \sqsubseteq \alpha.$$

Per definizione di $\sqsubseteq$ abbiamo che

$$a^n \sqsubseteq \alpha \implies \exists\, x \in V^* :\ \alpha = a^n x.$$

Essendo α la concatenazione di due stringhe finite, risulta che

$$\alpha = a^n x \wedge |a^n| = n \implies |\alpha| = n + |x|;$$

assumendo che $|x| = k$, possiamo affermare che $|\alpha| = n + k$.
Per concludere, essendo α un maggiorante di A^* in V^*, dovrà valere che $a^{n+k+1} \sqsubseteq \alpha$, ma questo è un assurdo dato che α non potrà contenere come sottostringa iniziale una stringa più lunga di $n + k$. □

ESERCIZIO 6.16

[1] Dimostrare che la funzione condizionale definita come segue

$$\mathtt{ifthenelse}_{ext}(x, y, z) = \begin{cases} \bot, & \text{se } x = \bot \\ \mathtt{ifthenelse}(x, y, z), & \text{altrimenti} \end{cases}$$

è una funzione continua.

4 Per la dimostrazione completa si veda la proposizione 6.6.3.1 presente in appendice.

Teorema 6.16 ($\mathtt{ifthenelse}_{ext}$ è continua)**.** *Sia* $\mathtt{ifthenelse} \equiv \lambda xyz.(x = 0) \to y, z$ *la funzione condizionale e sia*

$$\mathtt{ifthenelse}_{ext} \equiv \lambda xyz.(x = \bot) \to \bot, \mathtt{ifthenelse}(x, y, z)$$

la sua estensione su $\mathbb{NAT}^3$*. Possiamo provare che* $\mathtt{ifthenelse}_{ext}$ *è continua.*

Dimostrazione. Essendo che $\mathtt{ifthenelse}_{ext} : \mathbb{NAT}^3 \to \mathbb{NAT}$ e che in $\mathbb{NAT}^3$ possiamo avere unicamente catene finite lunghe al più 4, per il teorema 6.10 e 6.14 delle dispense del corso [1], risulta che

$$\mathtt{ifthenelse}_{ext} \text{ è continua } \iff \mathtt{ifthenelse}_{ext} \text{ è monotona.}$$

Di conseguenza, per mostrare che $\mathtt{ifthenelse}_{ext}$ è continua, ci basterà provare che è monotona e cioè che preserva l'ordinamento stabilito dalla relazione d'ordine $\sqsubseteq_3$ di $\mathbb{NAT}^3$:

$$\forall\, \vec{x}, \vec{y} \in \mathbb{NAT}^3,\ \vec{x} \sqsubseteq_3 \vec{y} \implies \mathtt{ifthenelse}_{ext}(\vec{x}) \sqsubseteq \mathtt{ifthenelse}_{ext}(\vec{y})$$

Consideriamo due qualsiasi $\vec{x}, \vec{y} \in \mathbb{NAT}^3$ tali che

$$\vec{x} = \langle x_1, x_2, x_3 \rangle \sqsubseteq_3 \langle y_1, y_2, y_3 \rangle = \vec{y}.$$

Per definizione di $\sqsubseteq_3$ sappiamo quindi che

$$x_1 \sqsubseteq y_1 \ \wedge\ x_2 \sqsubseteq y_2 \ \wedge\ x_3 \sqsubseteq y_3$$

Andiamo a lavorare per casi:

1. *caso* $x_1 = \bot$;
2. *caso* $x_1 = 0$;
3. *caso* $x_1 = n$ *con* $n > 0$.

1. *caso* $x_1 = \bot$. Per definizione di $\mathtt{ifthenelse}_{ext}$ ed essendo che $\bot$ è il minimo di $\mathbb{NAT}$, risulta che

   $$\mathtt{ifthenelse}_{ext}(\bot, x_2, x_3) = \bot \sqsubseteq \mathtt{ifthenelse}_{ext}(y_1, y_2, y_3).$$

2. *caso* $x_1 = 0$. Dato che $\mathbb{NAT}$ è un dominio piatto, abbiamo che

   $$0 \sqsubseteq y_1 \implies y_1 = 0.$$

   Per definizione di $\mathtt{ifthenelse}_{ext}$ e per ipotesi, possiamo affermare che

   $$\mathtt{ifthenelse}_{ext}(0, x_2, x_3) = x_2 \sqsubseteq y_2 = \mathtt{ifthenelse}_{ext}(0, y_2, y_3).$$

3. *caso* $x_1 = n$ *con* $n > 0$. Dato che $\mathbb{NAT}$ è un dominio piatto, abbiamo che

   $$n \sqsubseteq y_1 \implies y_1 = n.$$

   Per definizione di $\mathtt{ifthenelse}_{ext}$ e per ipotesi, possiamo affermare che

   $$\mathtt{ifthenelse}_{ext}(n, x_2, x_3) = x_3 \sqsubseteq y_3 = \mathtt{ifthenelse}_{ext}(n, y_2, y_3).$$

□

ESERCIZIO 7.6

[1] Fornire semantica operazionale e denotazionale del programma

letrec $\mathbf{f}(x) \Leftarrow \mathbf{f}(x)$ **in** $\mathbf{f}(5)$.

Riflettiamo anzitutto, a livello intuitivo, su quale sia il comportamento del programma **SLF** in questione: la funzione **f** è definita ricorsivamente e, indipendentemente dall'argomento fornitole al momento dell'invocazione, richiamerà sempre se stessa; l'esecuzione di $\mathbf{f}(5)$ genererà pertanto una sequenza infinita di chiamate ricorsive e il programma non terminerà mai.
Essendo che abbiamo a che fare con un programma divergente, ci aspettiamo che il suo valore semantico risulti indefinito:

- a livello operazionale, incapperemo in una sequenza infinita di riscritture;
- a livello denotazionale, otterremo come risultato della sua interpretazione semantica il valore $\perp$ di $\mathbb{NAT}$.

Andiamo a verificare formalmente quanto appena detto.

Esercizio 7.6.1 (*Semantica operazionale di* **letrec** $\mathbf{f}(x) \Leftarrow \mathbf{f}(x)$ **in** $\mathbf{f}(5)$)**.**
Sia $D = \{\mathbf{f}(x) \Leftarrow \mathbf{f}(x)\}$. Se prendiamo in considerazione la semantica operazionale con chiamata per nome di **SLF** [5] e proviamo ad assegnare un significato al nostro programma, otteniamo la seguente sequenza infinita di riscritture:

$$\mathbf{f}(5) \xrightarrow{(Fun)}_D \mathbf{f}(5) \xrightarrow{(Fun)}_D \mathbf{f}(5) \xrightarrow{(Fun)}_D \cdots$$

Anche se consideriamo la semantica operazionale con chiamata per valore di **SLF** [6], otteniamo una sequenza infinita di riscritture:

$$\mathbf{f}(5) \bullet\xrightarrow{(Fun')}_D \mathbf{f}(5) \bullet\xrightarrow{(Fun')}_D \mathbf{f}(5) \bullet\xrightarrow{(Fun')}_D \cdots$$

Esercizio 7.6.2 (*Semantica denotazionale di* **letrec** $\mathbf{f}(x) \Leftarrow \mathbf{f}(x)$ **in** $\mathbf{f}(5)$)**.**
Notiamo anzitutto che nel programma è stata dichiarata una sola funzione con arità 1, dunque, facendo riferimento alla definizione delle funzioni di interpretazione semantica fornite nella tabella 7.4 delle dispende del corso [1], risulterà che

$$\vec{F} = f, \qquad \pi_1 \vec{F} = f;$$
$$\vec{X} = x, \qquad \pi_1 \vec{X} = x.$$

5 Tabella 7.2 delle dispense del corso [1].
6 Tabella 7.3 delle dispense del corso [1].

Seguiamo un approccio bottom-up e precomputiamo le funzioni di interpretazione necessarie per dare la semantica del programma **SLF**:

$$\begin{aligned}
\mathcal{T}[\![5]\!] &= \lambda f.\lambda x.5 && \text{per } \textit{(Nat)}\\
\mathcal{T}[\![x]\!] &= \lambda f.\lambda x.x && \text{per } \textit{(Var)}\\
\mathcal{T}[\![\mathbf{f}(5)]\!] &= \lambda f.\lambda x.f(\mathcal{T}[\![5]\!]fx) && \text{per } \textit{(Fun)}\\
&= \lambda f.\lambda x.f((\lambda g.\lambda y.5)fx) && \text{per } \mathcal{T}[\![5]\!]\\
&= \lambda f.\lambda x.f(5) && \text{per } \beta\text{-riduzione}\\
\mathcal{T}[\![\mathbf{f}(x)]\!] &= \lambda f.\lambda x.f(\mathcal{T}[\![x]\!]fx) && \text{per } \textit{(Fun)}\\
&= \lambda f.\lambda x.f((\lambda g.\lambda y.y)fx) && \text{per } \mathcal{T}[\![x]\!]\\
&= \lambda f.\lambda x.f(x) && \text{per } \beta\text{-riduzione}\\
\mathcal{D}[\![\mathbf{f}(x) \Leftarrow \mathbf{f}(x)]\!] &= \mathbf{fix}(\lambda f.\mathcal{T}[\![\mathbf{f}(x)]\!]f) && \text{per } \textit{(Dec)}\\
&= \mathbf{fix}(\lambda f.(\lambda g.\lambda x.g(x))f) && \text{per } \mathcal{T}[\![f(x)]\!]\\
&= \mathbf{fix}(\lambda f.\lambda x.f(x)) && \text{per } \beta\text{-riduzione}
\end{aligned}$$

Essendo che il funzionale $\lambda f.\lambda x.f(x)$ è continuo, per il corollario 6.38 del teorema di Kleene [1], sappiamo che esso ha minimo punto fisso equivalente all'estremo superiore della catena di approssimanti della funzione **f**:

$$\mathcal{D}[\![\mathbf{f}(x) \Leftarrow \mathbf{f}(x)]\!] = \mathbf{sup}\{(\lambda f.\lambda x.f(x))^i\Omega \mid i \in \mathbb{N}\} \qquad \text{con } \Omega \equiv \lambda x.\bot$$

Calcolando le prime approssimanti della catena risulta semplice mostrare come questa sia convergente e abbia estremo superiore pari a Ω:

$$\begin{aligned}
(\lambda f.\lambda x.f(x))^0\Omega &= \Omega && \text{per } (\lambda f.\lambda x.f(x))^0 \equiv \mathrm{Id}_{\mathbb{F}\mathbb{U}\mathbb{N}}\\
(\lambda f.\lambda x.f(x))^1\Omega &= (\lambda f.\lambda x.f(x))((\lambda f.\lambda x.f(x))^0\Omega) && \text{per definizione della catena}\\
&= (\lambda f.\lambda x.f(x))\Omega && \text{per } (\lambda f.\lambda x.f(x))^0\Omega\\
&= \lambda x.\Omega(x) && \text{per } \beta\text{-riduzione}\\
&= \lambda x.\bot \equiv \Omega && \text{per } \beta\text{-riduzione}\\
(\lambda f.\lambda x.f(x))^2\Omega &= (\lambda f.\lambda x.f(x))((\lambda f.\lambda x.f(x))^1\Omega) && \text{per definizione della catena}\\
&= (\lambda f.\lambda x.f(x))\Omega && \text{per } (\lambda f.\lambda x.f(x))^1\Omega\\
&= \Omega && \text{per } (\lambda f.\lambda x.f(x))^1\Omega
\end{aligned}$$

Risulta quindi che

$$\mathcal{D}[\![\mathbf{f}(x) \Leftarrow \mathbf{f}(x)]\!] = \Omega$$

A questo punto abbiamo tutto quello che ci serve per dare la semantica denotazionale del programma **SLF**:

$$\begin{aligned}
\mathcal{P}[\![\mathbf{letrec}\ f(x) \Leftarrow f(x)\ \mathbf{in}\ f(5)]\!] &= \mathcal{T}[\![f(5)]\!]\mathcal{D}[\![f(x) \Leftarrow f(x)]\!]0 && \text{per } \textit{(Prg)}\\
&= (\lambda f.\lambda x.f(5))\Omega 0 && \text{per } \mathcal{T}[\![f(5)]\!] \text{ e } \mathcal{D}[\![f(x) \Leftarrow f(x)]\!]\\
&= (\lambda x.\Omega(5))0 && \text{per } \beta\text{-riduzione}\\
&= \Omega(5) && \text{per } \beta\text{-riduzione}\\
&= \bot && \text{per } \beta\text{-riduzione}
\end{aligned}$$

ESERCIZIO 8.5

[1] Si consideri il comando

c_1 **then when** e **do** c_2

con la seguente semantica informale: nello stato ottenuto eseguendo c_1 viene valutata la condizione e; se questa risulta vera, il comando c_2 viene valutato nello stato precedente l'esecuzione di c_1, altrimenti l'intero comando non ha effetto sullo stato. Fornire la semantica operazionale e denotazionale del comando sopra descritto.

Esercizio 8.5.1 (*Semantica operazionale del comando* c_1 **then when** e **do** c_2)**.**
Per eseguire il comando c_1 **then when** e **do** c_2 in uno stato σ, dovremo prima di tutto valutare c_1; tuttavia, non è detto che c_1 sia valutabile in un singolo passo, questo perchè la semantica operazionale dei comandi di **TINY** non è una semantica di valutazione. Se per ridurre c_1 in **noaction** fossero necessari n passi con $n \geqslant 2$, potremmo sfruttare la seguente regola per eseguire i primi $n-1$ passi del processo di valutazione:

$$\frac{\langle c_1, \sigma\rangle \longrightarrow \langle c_1', \sigma'\rangle \qquad c_1 \neq \textbf{noaction}}{\langle c_1 \textbf{ then when } e \textbf{ do } c_2, \sigma\rangle \longrightarrow \langle c_1' \textbf{ then when } e \textbf{ do } c_2, \sigma'\rangle} \quad (\mathit{ThenWhen}_0)$$

Se c_1 è stato ridotto in un comando valutabile in un singolo passo o lo è fin dall'inizio, potremo usare una di queste due regole per valutare completamente il comando principale:

$$\frac{\langle c_1, \sigma\rangle \longrightarrow \langle \textbf{noaction}, \sigma'\rangle \qquad \langle e, \sigma'\rangle \longrightarrow\!\!\!\!\!\rightarrow\ \sigma'' \qquad \sigma''(\mathtt{res}) = \text{true}}{\langle c_1 \textbf{ then when } e \textbf{ do } c_2, \sigma\rangle \longrightarrow \langle c_2, \sigma\rangle} \quad (\mathit{ThenWhen}_1)$$

$$\frac{\langle c_1, \sigma\rangle \longrightarrow \langle \textbf{noaction}, \sigma'\rangle \qquad \langle e, \sigma'\rangle \longrightarrow\!\!\!\!\!\rightarrow\ \sigma'' \qquad \sigma''(\mathtt{res}) = \text{false}}{\langle c_1 \textbf{ then when } e \textbf{ do } c_2, \sigma\rangle \longrightarrow \langle \textbf{noaction}, \sigma\rangle} \quad (\mathit{ThenWhen}_2)$$

Entrambe le regole codificano il fatto che l'espressione e verrà valutata nello stato σ' ottenuto a seguito del completamento della valutazione del comando c_1. Se e risultasse vera, potremmo usare la regola $(\mathit{ThenWhen}_1)$ ed il significato operazionale dell'intero comando corrisponderà a quello attribuito all'esecuzione di c_2 nello stato di partenza σ; altrimenti, sfrutteremo la regola $(\mathit{ThenWhen}_2)$ che ci permetterà di considerare il comando terminato e di ripristinare lo stato σ.

Esercizio 8.5.2 (*Semantica denotazionale del comando* c_1 **then when** e **do** c_2)**.**
Nel caso della semantica denotazionale dobbiamo considerare anche le possibili situazioni di errore; in particolare, il risultato dell'interpretazione dell'intero comando dovrà restituire errore nei seguenti casi:

- se l'interpretazione di c_1 nello stato σ ci dà errore;
- se l'interpretazione di e nello stato σ' ci dà errore;
- se il valore ottenuto dall'interpretazione di e non è un booleano.

Possiamo quindi aggiungere il seguente caso alla definizione della funzione di interpretazione dei comandi di **TINY**:

$$
\begin{array}{l}
\mathcal{C}[\![c_1 \textbf{ then when } e \textbf{ do } c_2]\!] \triangleq \lambda\sigma. \\
\quad \textbf{cases } \mathcal{C}[\![c_1]\!]\sigma \textbf{ of} \\
\quad\quad \sigma' : \\
\quad\quad\quad \textbf{cases } \mathcal{E}[\![e]\!]\sigma' \textbf{ of} \\
\quad\quad\quad\quad \langle \nu, \sigma'' \rangle : (\mathrm{isBOOL}\ \nu) \rightarrow \\
\quad\quad\quad\quad\quad\quad \nu \rightarrow \mathcal{C}[\![c_2]\!]\sigma, \sigma, \\
\quad\quad\quad\quad\quad\quad \mathit{error}; \\
\quad\quad\quad\quad \mathit{error} : \mathit{error}; \\
\quad\quad\quad \textbf{endcases}; \\
\quad\quad \mathit{error} : \mathit{error} \\
\quad \textbf{endcases}
\end{array}
$$

ESERCIZIO 9.9

> [1] Fornire una semantica non standard di **SMALL**, che ad ogni programma associa il numero corrispondente alle volte che un assegnamento viene effettuato dal programma.

Per fornire una semantica denotazionale di **SMALL** che ad ogni programma associa il numero di assegnamenti eseguiti, rispetto a quella vista durante il corso, dovremo andare a modificare la definizione della funzione di interpretazione semantica:

1. dei programmi, $\mathcal{P}$;
2. dei comandi, $\mathcal{C}$, nel caso specifico dell'assegnamento.

Inoltre, per tener traccia del numero di assegnamenti eseguiti, sfrutteremo una nuova locazione di memoria riservata che chiameremo `lassign`.

1. Il codominio della funzione di interpretazione dei programmi dovrà essere modificato in quanto, nel caso in cui non ci siano errori, il valore associato al programma non sarà più la sequenza di valori di base che rappresentano l'output, ma bensì un singolo naturale corrispondente al numero di assegnamenti eseguiti; risulta dunque che:

   $$\mathcal{P} : \ \mathrm{Prog} \longrightarrow \mathbb{BVAL}^* \longrightarrow (\mathbb{NAT} + \{\mathit{error}\}).$$

   La definizione di $\mathcal{P}$ dovrà quindi essere modificata per far sì che, in assenza di errori, al posto del contenuto di `lout` venga restituito quello di `lassign`; dovremo inoltre ricordarci di modificare la memoria iniziale $\sigma_0 \equiv \lambda l.\mathit{unused}$ in modo tale che alla locazione `lassign` venga inizialmente associato il valore 0:

   $$\mathcal{P}[\![\textbf{program } c]\!]\overrightarrow{in} \triangleq \mathcal{C}[\![c]\!]\rho_0\sigma_0[\overrightarrow{in}/\mathtt{lin}][\mathit{nil}/\mathtt{lout}][0/\mathtt{lassign}] \star \lambda\sigma.\sigma(\mathtt{lassign})$$

2. La funzione di interpretazione semantica dei comandi **SMALL** dovrà essere modificata in modo che, ogni volta in cui un assegnamento viene interpretato, il contenuto della locazione `lassign` venga incrementato di uno:

$$\mathcal{C}[\![e := e']\!]\rho \triangleq \mathcal{E}[\![e]\!]\rho \star \mathrm{check}\mathbb{LOC} \star \lambda l.\mathcal{R}[\![e']\!]\rho \star \lambda v\sigma.\sigma[v/l][\sigma(\texttt{lassign}) + 1/\texttt{lassign}]$$

3

SISTEMI CONCORRENTI

ESERCIZIO 11.8

[1] Dimostrare che l'unione di tutte le bisimulazioni di branching è:

1. una bisimulazione di branching;
2. un'equivalenza.

Per provare che l'unione di tutte le bisimulazioni di branching è una bisimulazione di branching, dobbiamo prima dimostrare il seguente lemma.

Lemma 11.8.1.1 (L'unione preserva le bisimulazioni di branching)**.**
Sia $\langle Q, A_\tau, \to \rangle$ *un LTS e sia ogni* S_i *con* $i \in \{1, \dots, n\}$ *una bisimulazione di branching su* Q*, possiamo provare che* $\bigcup_{i=1}^{n} S_i$ *è a sua volta una bisimulazione di branching.*

Dimostrazione. Vogliamo quindi mostrare che:

1. $\forall\, p, q \in Q,\ \langle p, q \rangle \in \bigcup_{i=1}^{n} S_i \implies \langle q, p \rangle \in \bigcup_{i=1}^{n} S_i$;
2. $\forall\, \langle p, q \rangle \in \bigcup_{i=1}^{n} S_i,\ \forall\, \mu \in A_\tau,\ \forall\, p' \in Q :\ p \xrightarrow{\mu} p'$, almeno una delle seguenti condizioni deve essere soddisfatta:
   a) $\mu = \tau \wedge \langle p', q \rangle \in \bigcup_{i=1}^{n} S_i$;
   b) $\exists\, q_x, q' \in Q :\ q \overset{\varepsilon}{\Rightarrow} q_x \xrightarrow{\mu} q'$ con $\langle p, q_x \rangle \in \bigcup_{i=1}^{n} S_i \wedge \langle p', q' \rangle \in \bigcup_{i=1}^{n} S_i$.

Consideriamo una qualunque coppia $\langle p, q \rangle \in \bigcup_{i=1}^{n} S_i$, per definizione di unione tra insiemi abbiamo che:

$$\langle p, q \rangle \in \bigcup_{i=1}^{n} S_i \implies \exists\, k \in \{1, \dots, n\} :\ \langle p, q \rangle \in S_k.$$

1. Essendo che S_k è simmetrica risulta

   $$\langle p, q \rangle \in S_k \implies \langle q, p \rangle \in S_k.$$

   Inoltre, dato che $S_k \subseteq \bigcup_{i=1}^{n} S_i$, possiamo affermare che

   $$\langle q, p \rangle \in S_k \implies \langle q, p \rangle \in \bigcup_{i=1}^{n} S_i;$$

2. Consideriamo un qualsiasi $p' \in Q :\ p \xrightarrow{\mu} p'$ con $\mu \in A_\tau$.

a) Se $\mu = \tau$, essendo che S_k è bisimulazione di branching, sappiamo che

$$\langle p, q\rangle \in S_k \ \wedge\ p \xrightarrow{\tau} p' \implies \langle p', q\rangle \in S_k;$$

Dato che $S_k \subseteq \bigcup_{i=1}^{n}$, risulta che

$$\langle p', q\rangle \in S_k \implies \langle p', q\rangle \in \bigcup_{i=1}^{n} S_i.$$

b) Se $\mu \neq \tau$, essendo che S_k è bisimulazione di branching, risulta che

$$\langle p, q\rangle \in S_k \ \wedge\ p \xrightarrow{\mu} p' \implies$$
$$\exists\, q_x, q' \in Q :\ q \stackrel{\varepsilon}{\Rightarrow} q_x \xrightarrow{\mu} q' \text{ con } \langle p, q_x\rangle \in S_k \ \wedge\ \langle p', q'\rangle \in S_k$$

Dato che $S_k \subseteq \bigcup_{i=1}^{n} S_i$, possiamo concludere che

$$\langle p, q_x\rangle \in S_k \implies \langle p, q_x\rangle \in \bigcup_{i=1}^{n} S_i$$
$$\langle p', q'\rangle \in S_k \implies \langle p', q'\rangle \in \bigcup_{i=1}^{n} S_i$$

□

Teorema 11.8.1 ($\approx_b$ è la più grande bisimulazione di branching)**.** *La bisimilarità di branching*

$$\approx_b \triangleq \bigcup \{R \mid R \textit{ è una bisimulazione di branching}\}$$

è una bisimulazione di branching.

Dimostrazione. Per il lemma 11.8.1.1 sappiamo che l'unione preserva le bisimulazioni di branching; di conseguenza, risulta immediato che l'unione di tutte le bisimulazioni di branching sia a sua volta una bisimulazione di branching. □

Per provare che l'unione di tutte le bisimulazioni di branching è un'equivalenza dovremo mostrare che questa è riflessiva, simmetrica e transitiva; per fare ciò dovremo, in ordine, sfruttare i risultati dei seguenti lemmi.

Lemma 11.8.2.1 (L'identità è una bisimulazione di branching)**.**
Sia $\langle Q, A_\tau, \to\rangle$ *un LTS e* Id_Q *la relazione identità su* Q*, possiamo provare che* Id_Q *è una bisimulazione di branching.*

Dimostrazione. Vogliamo quindi mostrare che:

1. Id_Q è simmetrica;
2. $\forall\ \langle p, p\rangle \in Id_Q,\ \forall\ \mu \in A_\tau,\ \forall\ p' \in Q :\ p \xrightarrow{\mu} p'$, almeno una delle seguenti condizioni deve essere soddisfatta:
   a) $\mu = \tau \ \wedge\ \langle p', p\rangle \in Id_Q$;
   b) $\exists\, q_x, q' \in Q :\ p \stackrel{\varepsilon}{\Rightarrow} q_x \xrightarrow{\mu} q'$ con $\langle p, q_x\rangle \in Id_Q \ \wedge\ \langle p', q'\rangle \in Id_Q$.

Consideriamo una qualunque coppia $\langle p, p\rangle \in Id_Q$.

1. Id_Q risulta simmetrico per definizione.

2. Consideriamo un qualsiasi $p' \in Q : \ p \xrightarrow{\mu} p'$ con $\mu \in A_\tau$.
   a) La prima condizione risulta soddisfatta solo nel caso in cui $p \equiv p'$, in tal caso avremo infatti che:

   $$\langle p, p\rangle \in Id_Q \ \wedge \ p \xrightarrow{\tau} p$$

   b) La seconda condizione risulta sempre soddisfatta, ci basta considerare $p \equiv q_x$ e $p' \equiv q'$ ed avremo che:

   $$\langle p, p\rangle \in Id_Q \ \wedge \ p \xrightarrow{\mu} p' \implies$$
   $$p \stackrel{\varepsilon}{\Rightarrow} p \xrightarrow{\mu} p' \text{ con } \langle p, p\rangle \in Id_Q \ \wedge \ \langle p', p'\rangle \in Id_Q$$

□

Lemma 11.8.2.2 (L'inversione preserva le bisimulazioni di branching)**.**
Sia $\langle Q, A_\tau, \rightarrow\rangle$ *un LTS e sia* S *una bisimulazione di branching su* Q*, possiamo provare che* S^{-1} *è a sua volta una bisimulazione di branching* [1].

Dimostrazione. Vogliamo quindi mostrare che:

1. $\forall\, q, p \in Q, \ \langle q, p\rangle \in S^{-1} \implies \langle p, q\rangle \in S^{-1}$;

2. $\forall \langle q, p\rangle \in S^{-1} \, \forall \mu \in A_\tau, \ \forall q' \in Q : \ q \xrightarrow{\mu} q'$, almeno una delle seguenti condizioni deve essere soddisfatta:
   a) $\mu = \tau \ \wedge \ \langle q', p\rangle \in S^{-1}$;
   b) $\exists\, p_x, p' \in Q : \ p \stackrel{\varepsilon}{\Rightarrow} p_x \xrightarrow{\mu} p'$ con $\langle q, p_x\rangle \in S^{-1} \ \wedge \ \langle q', p'\rangle \in S^{-1}$.

Consideriamo una qualunque coppia $\langle q, p\rangle \in S^{-1}$, per definizione di inversa abbiamo che

$$\langle q, p\rangle \in S^{-1} \implies \langle p, q\rangle \in S.$$

Essendo che S è per ipotesi simmetrica, sappiamo anche che

$$\langle p, q\rangle \in S \implies \langle q, p\rangle \in S.$$

1. Per definizione di inversa sappiamo che

   $$\langle q, p\rangle \in S \implies \langle p, q\rangle \in S^{-1}.$$

2. Consideriamo un qualsiasi $q' \in Q : \ q \xrightarrow{\mu} q'$ con $\mu \in A_\tau$.

1 *Dimostrazione alternativa.* Dato che S è bisimulazione di branching, sappiamo che S è simmetrica; inoltre, per definizione di relazione simmetrica, abbiamo che $S = S^{-1}$ e, per tale motivo, possiamo concludere che anche S^{-1} è bisimulazione di branching.

a) Se $\mu = \tau$, essendo che S è bisimulazione di branching, sappiamo per certo che

$$\langle q, p\rangle \in S \ \wedge \ q \xrightarrow{\tau} q' \implies \langle q', p\rangle \in S;$$

Dato che S è simmetrica, risulta che

$$\langle q', p\rangle \in S \implies \langle p, q'\rangle \in S.$$

Per definizione di inversa, concludiamo che

$$\langle p, q'\rangle \in S \implies \langle q', p\rangle \in S^{-1}.$$

b) Se $\mu \neq \tau$, essendo che S è bisimulazione di branching, risulta che

$$\langle q, p\rangle \in S \ \wedge \ q \xrightarrow{\mu} q' \implies$$
$$\exists\, p_x, p' \in Q : \ p \overset{\varepsilon}{\Rightarrow} p_x \xrightarrow{\mu} p' \text{ con } \langle q, p_x\rangle \in S \ \wedge \ \langle q', p'\rangle \in S$$

Dato che S è simmetrica, sappiamo che

$$\langle q, p_x\rangle \in S \implies \langle p_x, q\rangle \in S;$$
$$\langle q', p'\rangle \in S \implies \langle p', q'\rangle \in S.$$

Per definizione di inversa, possiamo concludere che

$$\langle p_x, q\rangle \in S \implies \langle q, p_x\rangle \in S^{-1};$$
$$\langle p', q'\rangle \in S \implies \langle q', p'\rangle \in S^{-1}.$$

□

Lemma 11.8.2.3 (La composizione preserva le bisimulazioni di branching)**.**
Sia $\langle Q, A_\tau, \to\rangle$ un LTS e siano S_1 e S_2 due bisimulazioni di branching su Q, possiamo provare che $S_1 \cdot S_2$ è a sua volta una bisimulazione di branching.

Dimostrazione. Diversamente da quanto fatto precedentemente [2], per provare che $S_1 \cdot S_2$ è una bisimulazione di branching faremo vedere che $\forall\, \langle p, r\rangle \in S_1 \cdot S_2$, $\forall\, \mu \in A_\tau$ risulta che:

1. r *branching simula* p: $\forall\, p' \in Q : \ p \xrightarrow{\mu} p'$, almeno una delle seguenti condizioni deve essere soddisfatta:
   a) $\mu = \tau \ \wedge \ \langle p', r\rangle \in S_1 \cdot S_2$;
   b) $\exists\, r_x, r' \in Q : \ r \overset{\varepsilon}{\Rightarrow} r_x \xrightarrow{\mu} r'$ con $\langle p, r_x\rangle \in S_1 \cdot S_2 \ \wedge \ \langle p', r'\rangle \in S_1 \cdot S_2$.
2. p *branching simula* r: $\forall\, r' \in Q : \ r \xrightarrow{\mu} r'$, almeno una delle seguenti condizioni deve essere soddisfatta:
   a) $\mu = \tau \ \wedge \ \langle r', p\rangle \in S_1 \cdot S_2$;
   b) $\exists\, p_x, p' \in Q : \ p \overset{\varepsilon}{\Rightarrow} p_x \xrightarrow{\mu} p'$ con $\langle r, p_x\rangle \in S_1 \cdot S_2 \ \wedge \ \langle r', p'\rangle \in S_1 \cdot S_2$.

2 Sapere che S_1 e S_2 sono simmetriche non basta a mostrare che $S_1 \cdot S_2$ è a sua volta simmetrica; per mostrare ciò dovremmo anche sapere che $S_1 \cdot S_2 = S_2 \cdot S_1$.

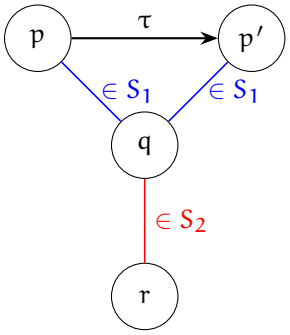

Figura 3: Composizione di bisimulazioni di branching, caso a).

1. Consideriamo una qualunque coppia $\langle p, r\rangle \in S_1 \cdot S_2$ ed un qualsiasi $p' \in Q : \ p \xrightarrow{\mu} p'$ con $\mu \in A_\tau$. Per definizione di composizione, abbiamo che

$$\langle p, r\rangle \in S_1 \cdot S_2 \implies \exists\, q \in Q : \ \langle p, q\rangle \in S_1 \ \wedge \ \langle q, r\rangle \in S_2.$$

   a) Se $\mu = \tau$, essendo che S_1 è bisimulazione di branching, sappiamo per certo che

$$\langle p, q\rangle \in S_1 \ \wedge \ p \xrightarrow{\tau} p' \implies \langle p', q\rangle \in S_1.$$

   Per definizione di composizione, possiamo concludere che

$$\langle p', q\rangle \in S_1 \ \wedge \ \langle q, r\rangle \in S_2 \implies \langle r', p\rangle \in S_1 \cdot S_2.$$

   Graficamente abbiamo la situazione illustrata nella figura 3.

   b) Se $\mu \neq \tau$, essendo che S_1 è bisimulazione di branching, risulta che

$$\langle p, q\rangle \in S_1 \ \wedge \ p \xrightarrow{\mu} p' \implies$$
$$\exists\, q_x, q' \in Q : \ q \stackrel{\varepsilon}{\Rightarrow} q_x \xrightarrow{\mu} q' \text{ con } \langle p, q_x\rangle \in S_1 \ \wedge \ \langle p', q'\rangle \in S_2$$

   Per definizione di relazione di transizione debole, sappiamo che

$$q \stackrel{\varepsilon}{\Rightarrow} q_x \implies \exists\, n \in \mathbb{N} : \ q(\xrightarrow{\tau})^n q_x.$$

   Lavorando per induzione su n, lunghezza della sequenza di azioni invisibili eseguite a partire da q per arrivare in q_x, potremmo facilmente dimostrare [3] che

$$\langle q, r\rangle \in S_2 \ \wedge \ q \stackrel{\varepsilon}{\Rightarrow} q_x \implies \langle q_x, r\rangle \in S_2.$$

   A questo punto sappiamo che $\langle q_x, r\rangle \in S_2$ e che $q_x \xrightarrow{\mu} q'$ con $\mu \neq \tau$, ma allora, essendo che S_2 è bisimulazione di branching, risulta che

$$\langle q_x, r\rangle \in S_2 \ \wedge \ q_x \xrightarrow{\mu} q' \implies$$
$$\exists\, r_x, r' \in Q : \ r \stackrel{\varepsilon}{\Rightarrow} r_x \xrightarrow{\mu} r' \text{ con } \langle q_x, r_x\rangle \in S_2 \ \wedge \ \langle q', r'\rangle \in S_2.$$

   Per definizione di composizione di relazioni, possiamo quindi concludere che

$$\langle p, q_x\rangle \in S_1 \ \wedge \langle q_x, r_x\rangle \in S_2 \implies \langle p, r_x\rangle \in S_1 \cdot S_2$$
$$\langle p', q'\rangle \in S_1 \ \wedge \ \langle q', r'\rangle \in S_2 \implies \langle p', r'\rangle \in S_1 \cdot S_2.$$

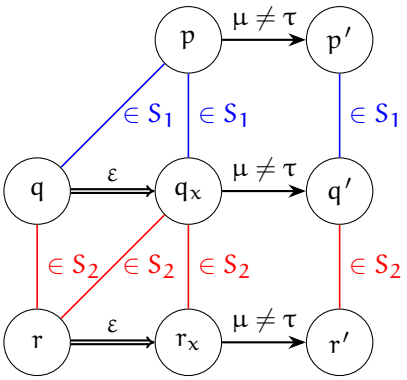

Figura 4: Composizione di bisimulazioni di branching, caso b).

Graficamente abbiamo la situazione illustrata nella figura 4.

2. Per provare che p *branching simula* r procediamo in modo totalmente identico a quanto fatto per provare che r *branching simula* p.

□

Teorema 11.8.2 ($\approx_b$ è un'equivalenza)**.** *La bisimilarità di branching $\approx_b$ è una relazione d'equivalenza.*

Dimostrazione. Vogliamo quindi provare che $\approx_b$ è:

1. riflessiva: $\forall\, p \in Q,\ p \approx_b p$;
2. simmetrica: $\forall\, p, q \in Q,\ p \approx_b q \implies q \approx_b p$;
3. transitiva: $\forall\, p, q, r \in Q,\ p \approx_b q \land q \approx_b r \implies p \approx_b r$.

1. *Proviamo che $\approx_b$ è riflessiva*. Per il lemma 11.8.2.1 sappiamo che Id_Q è una bisimulazione di branching; di conseguenza, per definizione di $\approx_b$ risulta che $Id_Q \subseteq \approx_b$. Possiamo allora concludere che

$$\forall\, p \in Q,\ \langle p, p\rangle \in Id_Q \subseteq \approx_b \implies p \approx_b p.$$

2. *Proviamo che $\approx_b$ è simmetrica*. Per definizione di $\approx_b$, risulta che

$$\forall\, p, q \in Q : p \approx_b q,\ \exists \text{ bisimulazione di branching } S : \langle p, q\rangle \in S.$$

Per definizione di inversa, abbiamo che

$$\langle p, q\rangle \in S \implies \langle q, p\rangle \in S^{-1}.$$

Per il lemma 11.8.2.2 sappiamo che anche S^{-1} è bisimulazione di branching e, dunque, per definizione di $\approx_b$ risulta che $S^{-1} \subseteq \approx_b$. Possiamo quindi concludere che

$$\langle q, p\rangle \in S^{-1} \subseteq \approx_b \implies q \approx_b p.$$

3 Per la dimostrazione completa si veda la proposizione 11.8.2.1 presente in appendice.

3. *Proviamo che $\approx_b$ è transitiva*. Consideriamo $p, q, r \in Q : \ p \approx_b q \ \wedge \ q \approx_b r$. Per definizione di $\approx_b$, risulta che

$$p \approx_b q \implies \exists \text{ bisimulazione di branching } S_1 : \langle p, q\rangle \in S_1;$$
$$q \approx_b r \implies \exists \text{ bisimulazione di branching } S_2 : \langle q, r\rangle \in S_2.$$

Per definizione di composizione di relazioni, abbiamo che

$$\langle p, q\rangle \in S_1 \ \wedge \ \langle q, r\rangle \in S_2 \implies \langle p, r\rangle \in S_1 \cdot S_2.$$

Per il lemma 11.8.2.3 sappiamo che anche $S_1 \cdot S_2$ è bisimulazione di branching e, dunque, per definizione di $\approx_b$ risulta che $S_1 \cdot S_2 \subseteq \approx_b$. Possiamo quindi concludere che

$$\langle p, r\rangle \in S_1 \cdot S_2 \subseteq \approx_b \implies p \approx_b r.$$

□

A

APPENDICE: DIMOSTRAZIONI ADDIZIONALI

Proposizione 6.6.3.1. *Sia* $(V^*, \sqsubseteq)$ *il poset definito nel teorema 6.6.1 e sia* A^* *l'insieme delle stringhe finite sull'alfabeto* $A = \{a\}$*. Possiamo provare che* $(A^*, \sqsubseteq)$ *è una catena in* $(V^*, \sqsubseteq)$ [1].

Dimostrazione. Gli elementi di A^* possono essere messi in corrispodenza biunivoca con i naturali nel seguente modo:

$$a^0 = \varepsilon;$$
$$a^{n+1} = aa^n = a^n a \;\forall\, n \geqslant 0.$$

Di conseguenza, per provare che $(A^*, \sqsubseteq)$ è una catena in $(V^*, \sqsubseteq)$, dobbiamo mostrare che

$$\forall\, n \in \mathbb{N},\; a^n \in V^* \;\wedge\; a^n \sqsubseteq a^{n+1}.$$

Procediamo nella dimostrazione lavorando per induzione su n.

- *Caso base* ($n = 0$). Essendo che $a^0 = \varepsilon$ e che V^* è chiusura riflessiva e transitiva di V, risulta che $a^0 \in V^*$. Dato che ε è l'elemento neutro della concatenazione di stringhe, abbiamo che $a = \varepsilon a$; dunque, per definizione di $\sqsubseteq$, possiamo concludere che $a^0 = \varepsilon \sqsubseteq a = a^1$.

- *Passo induttivo* ($n = k + 1$). Assumiamo che la tesi sia vera per stringhe lunghe al più k e mostriamo che risulta ancora vera per stringhe lunghe $k+1$. Per definizione di concatenazione di stringhe, abbiamo che

  $$a^{k+1} = aa^k.$$

  Per ipotesi induttiva ed essendo che $\forall\, n \in \mathbb{N},\; |a^n| = n$, sappiamo che

  $$a \in V \subseteq V^* \;\wedge\; a \in V^k \subseteq V^*.$$

  Dunque, per concatenazione di linguaggi e per definizione di V^*, possiamo affermare che

  $$aa^k \in V \cdot V^k = V^{k+1} \subseteq V^*.$$

  Infine, essendo che $a^{k+2} = a^{k+1}a$, per definizione di $\sqsubseteq$, possiamo concludere che $a^{k+1} \sqsubseteq a^{k+2}$.

□

1 Questa proposizione è stata usata nella prova del teorema 6.6.3.

Proposizione 11.8.2.1. *Sia* $\langle Q, A_\tau, \to \rangle$ *un LTS e sia* S *una bisimulazione di branching su* Q*, possiamo provare che* [2]

$$\forall\, q, q_x, r \in Q,\ \langle q, r\rangle \in S \ \wedge\ q \overset{\varepsilon}{\Rightarrow} q_x \implies \langle q_x, r\rangle \in S.$$

Dimostrazione. Per definizione di relazione di transizione debole, sappiamo che

$$q \overset{\varepsilon}{\Rightarrow} q_x \implies \exists\, n \in \mathbb{N} :\ q(\xrightarrow{\tau})^n q_x;$$

di conseguenza, per dimostrare quanto desiderato, possiamo lavorare per induzione su n, lunghezza della sequenza di τ eseguiti a partire da q per arrivare in q_x.

- *Caso base* ($n = 0$).. Per definizione di potenza di una relazione sappiamo che $(\xrightarrow{\tau})^0 = Id_Q$, ma allora possiamo affermare che

  $$q(\xrightarrow{\tau})^0 q_x \implies q \equiv q_x.$$

  Essendo che per ipotesi sappiamo già che $\langle q, r\rangle \in S$, risulta immediato che

  $$\langle q_x, r\rangle \in S.$$

- *Passo induttivo* ($n = k + 1$). Assumiamo che la tesi sia vera per sequenze di azione τ lunghe al più k e mostriamo che risulta ancora vera per sequenze lunghe $k + 1$. Per definizione di potenza di una relazione sappiamo che $(\xrightarrow{\tau})^{k+1} = (\xrightarrow{\tau}) \cdot (\xrightarrow{\tau})^k$, ma allora possiamo affermare che

  $$q(\xrightarrow{\tau})^{k+1} q_x \implies \exists\, q_w :\ q(\xrightarrow{\tau}) q_w (\xrightarrow{\tau})^k q_x.$$

  Per ipotesi induttiva, risulta che

  $$\langle q, r\rangle \in S \ \wedge\ q(\xrightarrow{\tau}) q_w \implies \langle q_w, r\rangle \in S.$$

  Sempre per ipotesi induttiva, possiamo infine concludere che

  $$\langle q_w, r\rangle \in S \ \wedge\ q_w(\xrightarrow{\tau})^k q_x \implies \langle q_x, r\rangle \in S.$$

□

2 Questa proposizione è stata usata nella prova del lemma 11.8.2.3.

BIBLIOGRAFIA

[1] Rosario Pugliese: *Note per il corso di Modelli di Sistemi Sequenziali e Concorrenti*, 2016. Versione del 3 novembre 2016. (Cited on pages 1, 4, 5, 6, 8, 9, 11, 12, 13, 14, 15, 16, and 18.)