```
 1
 2/* TemaEsame_Schema_Inventato3: Esercizio 5 */
 3/* Logica Asmeta */
 4
 5asm scommesse
 6
 7import StandardLibrary
 8import CTLlibrary
 9import LTLlibrary
10
11signature:
12    enum domain ValoriScommessa = {UNO, DUE, PARI}
13    domain NumPartita subsetof Integer
14
15    // budget dell'utente
16    // dato che sono numeri interi positivi dovrei
17    // utilizzare Natural, ma se lo faccio Asmeta da errore...
18    controlled budget : NumPartita -> Integer
19
20    // scommesse fatte su ogni partita dall'utente
21    // data una partita e una scommessa fatta dall'utente, restituisce
22    // la scommessa che l'utente ha fatto
23    monitored scommessa : NumPartita -> ValoriScommessa
24
25    // risultati delle partite monitored poiché altrimenti
26    // non riesco a fare lo scenario avalla con la choose rule
27    monitored risultati : NumPartita -> ValoriScommessa
28
29    // derivata che fa la somma di tutti i budget
30    // e dice se è stato raggiunto 50 (true)
31    derived check50 : Boolean
32
33    // derivata che dice se il budget di OGNI partita
34    // è >=1€ (true)
35    // (se ho 2€ posso ancora permettermi di
36    // perdere al massimo 2€, se ho 1€ no)
37    derived check1 : Boolean
38
39    // derivata che controlla il budget passato come
40    // argomento sia 0
41    derived check0 : NumPartita -> Boolean
42
43    // derivata che calcola il budget totale
44    derived totale : Integer
45
46
47definitions:
48    domain NumPartita = {1:5}
49
50    function check50 =
51        (budget(1)+budget(2)+budget(3)+budget(4)+budget(5)>=50)
52
53    function check1 =
54        ( forall $b in NumPartita with budget($b) > 1 )
55
56    function check0($num in NumPartita) =
57        budget($num)=0
```

```
58
59      function totale =
60          budget(1)+budget(2)+budget(3)+budget(4)+budget(5)
61
62      //-------------------------------------------------------
63      // PROPRIETA':
64      // 1) esiste un cammino in cui il giocatore arriva a 50
65      CTLSPEC ef(check50)
66
67      // 2) è possibile che sia azzerato il budget
68      // la interpreto come: "è possibile che uno dei budget vada a 0"
69      CTLSPEC ef( check0(1) or check0(2) or check0(3) or check0(4) or check0(5) )
70
71      // 3) non puoi mai perdere tutto
72      CTLSPEC not ef(totale=0)
73
74      // PROPRIETA' MIE:
75      // proprietà falsa e per cui in controesempio è utile:
76      // 1') non è mai possibile arrivare ad un budget >=50€
77      CTLSPEC not ef(check50)
78
79      //-------------------------------------------------------
80
81      main rule r_Main =
82          // se non ho raggiunto 50€ (not check50) e ho ancora
83          // soldi per giocare (check1) allora continuo
84          if (not check50) and check1 then
85              forall $num in NumPartita with true do
86                  // se indovina il risultato vince 1€
87                  if(scommessa($num)=risultati($num)) then
88                      budget($num) := budget($num)+1
89                  // se non ha indovinato si esegue il ramo else
90                  else
91                      // se c'è stato un pareggio perde 1€
92                      if (risultati($num) = PARI) then
93                          budget($num) := budget($num)-1
94                      // negli altri casi perde 2€
95                      else
96                          budget($num) := budget($num)-2
97                      endif
98                  endif
99          endif
100
101
102 default init s0:
103     // all'inizio il budget per ogni partita è di 5€
104     function budget($b in NumPartita) = 5
105
106
```

ATTENZIONE: **NuSMV non funziona**. Sospetto sia dovuto al "forall" che c'è nel main.

```
/* TemaEsame_Schema_Inventato3: Esercizio 6 */
/* Scenario Avalla */

scenario scenarioScommesse

load scommesse.asm

check budget(1) = 5;
check budget(2) = 5;
check budget(3) = 5;
check budget(4) = 5;
check budget(5) = 5;

/* Scenario in cui scommetto 1 sola volta e:
 * 1 - perdo 2€
 * 2 - vinco 1€
 * 3 - perdo 1€
 * 4 - vinco 1€
 * 5 - vinco 1€
 */

set scommessa(1) := UNO;
set risultati(1) := DUE;

set scommessa(2) := UNO;
set risultati(2) := UNO;

set scommessa(3) := UNO;
set risultati(3) := PARI;

set scommessa(4) := UNO;
set risultati(4) := UNO;

set scommessa(5) := UNO;
set risultati(5) := UNO;

step
check budget(1) = 3;
check budget(2) = 6;
check budget(3) = 4;
check budget(4) = 6;
check budget(5) = 6;
```

ATTENZIONE: i check non passano anche se nella simulazione i risultati sono corretti. I check non passano nemmeno se si esporta lo scenario avalla costruito automaticamente.

Sospetto sia dovuto al "forall" che c'è nel main.