

Sistemi a Tempo Reale

Informatica - Tiziano Villa

8 Gennaio 2008

Nome e Cognome:

Matricola:

Posta elettronica:

problema	punti massimi	i tuoi punti
problema 1	10	
problema 2	10	
problema 3	10	
totale	30	

1. (a) Si presenti l'algoritmo di minimizzazione di macchine a stati finiti deterministiche.

Traccia di risposta.

Si veda Lez. 13, p. 20, p. 22.

- (b) Si presenti un algoritmo per verificare se due macchine a stati finiti deterministiche sono equivalenti, se ne giustifichi la correttezza (enunciando il teorema su cui si basa) e se ne discuta la complessità'.

Traccia di risposta.

Si utilizzano i seguenti teoremi

- Due macchine a stati finiti deterministiche M_1 e M_2 sono equivalenti se e solo se c'è una bisimulazione tra M_1 e M_2 .
- C'è una bisimulazione tra due macchine a stati finiti deterministiche M_1 e M_2 se e solo se c'è un isomorfismo tra $\min(M_1)$ e $\min(M_2)$. [Si noti che $\min(M)$ produce la macchina unica con meno stati bisimile a M]

Ne risulta l'algoritmo:

- i. Minimizzare M_1 ottenendo $\min(M_1)$;
- ii. Minimizzare M_2 ottenendo $\min(M_2)$;
- iii. Verificare se $\min(M_1)$ e $\min(M_2)$ sono identiche a meno di ridenominazione degli stati (versione semplificata dell'isomorfismo tra grafi perché ci sono stati iniziali e etichette sui lati che guidano la verifica dell'isomorfismo).

La complessità' della minimizzazione degli stati di una macchina a stati finiti è quadratica nel numero degli stati (l'algoritmo più efficiente è dell'ordine $O(n \log n)$) e lineare nella cardinalità dell'insieme degli ingressi $|I|$. Questa versione ristretta dell'isomorfismo tra grafi è lineare nel numero degli stati (si noti che è necessario che $\min(M_1)$ e $\min(M_2)$ abbiano lo stesso numero di stati affinché siano isomorfe).

[La procedura comunemente presentata nei libri di testo di reti logiche per minimizzare macchine completamente specificate usando una tabella a scala ha complessità' $O(n^4|I|)$: la costruzione della tabella iniziale richiede $O(n^2|I|)$ operazioni (confronto di $(n-1)n/2$ coppie di stati ciascuna per $|I|$ ingressi), ogni iterazione richiede $O(n^2|I|)$ operazioni (esame di $(n-1)n/2$ caselle che contengono al più $|I|$ condizioni) e ci sono al più $O(n^2)$ iterazioni ($(n-1)n/2$ iterazioni se l'unica croce contenuta nella casella iniziale si propaga a tutte le altre caselle un passo per iterazione). Analisi di Luccio-Pagli]

(c) Si considerino le due macchine a stati finiti seguenti:

Macchina M' :

- stati: $s'_a, s'_b, s'_c, s'_d, s'_e$ con s'_a stato iniziale;
- transizione da s'_a a s'_b : $\bullet/0$,
transizione da s'_a a s'_c : $\bullet/0$,
transizione da s'_b a s'_d : $\bullet/0$,
transizione da s'_c a s'_e : $\bullet/1$,
transizione da s'_d a s'_d : $\bullet/0$,
transizione da s'_e a s'_e : $\bullet/0$.

Macchina M'' :

- stati: $s''_x, s''_y, s''_z, s''_u$ con s''_x stato iniziale;
- transizione da s''_x a s''_y : $\bullet/0$,
transizione da s''_y a s''_z : $\bullet/0$,
transizione da s''_y a s''_u : $\bullet/1$,
transizione da s''_z a s''_z : $\bullet/0$,
transizione da s''_u a s''_u : $\bullet/0$.

Si risponda in ordine alle seguenti domande (si indichi sempre il numerale romano in ogni risposta):

- i. Si disegnino i diagrammi di transizione delle due macchine.
- ii. Si classifichino le macchine rispetto al determinismo.
Traccia di risposta.
 M' e' nondeterministica, ma non pseudo-nondeterministica.
 M'' e' pseudo-nondeterministica.
- iii. Si derivino i comportamenti (successioni d'ingressi/successioni d'uscite) prodotti dalle due macchine e li si confrontino.
Traccia di risposta.
Per descrivere i comportamenti si possono usare le espressioni regolari.
 $\text{Comportamenti}(M') = (\bullet 0)^* + (\bullet 0)(\bullet 1)(\bullet 0)^*.$
 $\text{Comportamenti}(M'') = (\bullet 0)((\bullet 0)^* + (\bullet 1)(\bullet 0)^*).$
Dall'algebra delle espressioni regolari si vede che
 $\text{Comportamenti}(M') = \text{Comportamenti}(M'').$
- iv. Si trovi una simulazione di M' da parte di M'' , se esiste.
Traccia di risposta.
 M' e' simulata da M'' come mostrato dalla relazione

$$R_{M'-M''} = \{(s'_a, s''_x), (s'_b, s''_y), (s'_c, s''_y), (s'_d, s''_z), (s'_e, s''_u)\}.$$

- v. Si trovi una simulazione di M'' da parte di M' , se esiste.

Traccia di risposta.

M'' non e' simulata da M' perche' s''_x dovrebbe essere simulato da s'_a e quindi a sua volta ci dovrebbe essere uno stato di M' che simula s'_y , ma tale stato non esiste (ne' s'_b , ne' s'_c simulano s'_y perche' nessuno dei due ha sia una transizione con uscita 1 sia una transizione con uscita 0).

- vi. Si trovi una bisimulazione tra le due macchine, se esiste.

Traccia di risposta.

Poiche' M'' non e' simulata da M' non ci puo' essere una bisimulazione tra le due macchine.

- vii. Si commentino i risultati precedenti.

Traccia di risposta.

Le due macchine M' e M'' sono un esempio di macchine a stati finiti nondeterministiche equivalenti e ciascuna minimizzata (nella classe delle macchine bisimili), ma non isomorfe; in altri termini, esse mostrano che non esiste un'unica macchina a stati finita nondeterministica che realizza il sistema originale con il minimo numero di stati.

Si ricordi che se M' e' simulata da M'' allora M' raffina M'' . In generale non vale il viceversa, cioe' per M' e M'' nondeterministiche il fatto che M'' raffini M' non implica che ci sia una simulazione di M'' da parte di M' (ad es. nel nostro caso non c'e'). Ma se M' nondeterministica raffina M'' pseudo-nondeterministica allora esiste una simulazione di M' da parte di M'' (e infatti nel nostro caso c'e').

2. Si consideri il seguente automa temporizzato:

- locazioni: l_1, l_2 , dove l_1 e' la locazione iniziale con condizioni iniziali $s(0) := 0$;
- dinamica della locazione l_1 : $\dot{s}(t) = 1, y(t) = -s(t)$,
dinamica della locazione l_2 : $\dot{s}(t) = -1, y(t) = s(t)$;
- transizione da l_1 a l_2 : $A/assente, s(t) := 1$,
transizione da l_1 a l_2 : $B/assente, s(t) := 2$,
transizione da l_2 a l_2 : $A/assente, s(t) := s(t) + 1$,
transizione da l_2 a l_2 : $B/assente, s(t) := s(t) + 2$,
transizione da l_2 a l_1 : $C/s(t), s(t) := s(t)$,
dove $A = \{(u(t), s(t)) \mid u(t) = a\}$,
dove $B = \{(u(t), s(t)) \mid u(t) = b\}$,
dove $C = \{(u(t), s(t)) \mid u(t) = assente \wedge s(t) = 0\}$
(la sintassi delle annotazioni di una transizione e' *guardia/uscita, azione*);
- ingresso $u(t) \in \{a, b, assente\}$;
- uscita $y(t) \in Reali \cup \{assente\}$.

(a) Si supponga che l'ingresso $u(t)$ sia dato da

$$\forall t \in \mathbf{Reali}, u(t) = \begin{cases} a & \text{se } t = 1 \\ b & \text{se } t = 2 \\ \text{assente} & \text{altrimenti.} \end{cases}$$

Si disegni il diagramma di transizione degli stati dell'automa, annotando con precisione locazioni e transizioni.

- (b) Si disegni sugli assi delle coordinate (con il tempo in ascissa) la variabile di stato $s(t)$ e quella d'uscita $y(t)$ nell'intervallo $t \in [0, 6]$.

Traccia di risposta.

I grafici dello stato $s(t)$ e dell'uscita $y(t)$ sono tracciati in Fig. 1.

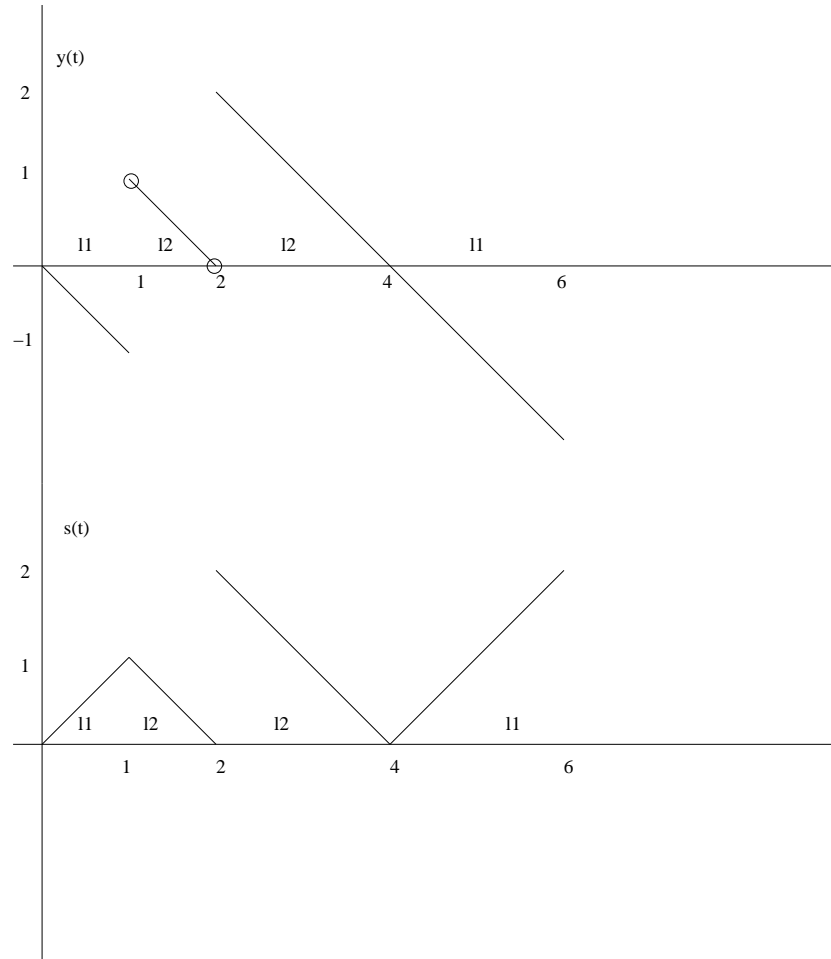


Figure 1: In basso il tracciato di $s(t)$ e in alto il tracciato di $y(t)$. Si noti che per $t = 1$ e $t = 2$ l'uscita $y(t)$ e' assente.

3. (a) Si argomenti l'ottimalita' dell'algoritmo di assegnamento RMS (Rate Monotonic Scheduling) per il caso di 2 processi.

Traccia di risposta.

La tesi che l'algoritmo di assegnamento RMS sia ottimale significa che se esiste una schedulazione valida con un algoritmo a priorit  fissata, anche RMS produrr  una schedulazione valida (cio  che soddisfa i periodi).

Si supponga che i processi $P_1 = (c_1, p_1)$ e $P_2 = (c_2, p_2)$ (c_i durata del processo, p_i periodo del processo) possano essere schedulati con un algoritmo a priorit  fissata che non sia RMS per il quale P_2 abbia priorit  maggiore e quindi sia $p_1 < p_2$ (perch  non sia RMS). Ci si ponga nel caso peggiore per il processo a priorit  minore, che   quello in cui entrambi i processi siano pronti per l'esecuzione allo stesso tempo (cio  i periodi di entrambi all'inizio partono insieme). Tale caso produce il tempo di risposta maggiore per il processo a priorit  minore. Con l'ipotesi precedente affinche' entrambi i processi siano schedulabili   necessario che sia $c_1 + c_2 \leq p_1$, altrimenti P_1 mancherebbe il suo periodo. Grazie a questa relazione tra tempo di calcolo e periodo di P_1 , si possono schedulare i processi anche rovesciando le priorit  e quindi eseguendo per primo P_1 come sarebbe per l'algoritmo RMS.

- (b) Dati i seguenti processi che eseguono su una unita' di calcolo singola, si mostri un assegnamento secondo l'algoritmo RMS (Rate Monotonic Scheduling), se esiste.

processo	tempo di esecuzione	scadenza
P1	2	4
P2	1	12

Si calcoli l'utilizzo dell'unita' di calcolo da parte di questi processi.

Traccia di risposta.

```
P1 X X - - | X X - - | X X - - |
P2 - - X - - - - - - - - - |
```

L'utilizzo dell'unita' di calcolo e' $U = \sum(c_i/p_i) = 2/4 + 1/12 = 7/12 \sim 0,583$ che e' ben al di sotto del maggiorante $m \cdot (2^{1/m} - 1) = 2 \cdot (2^{1/2} - 1) \sim 0,83$. Essa e' solo una condizione sufficiente, verificata in questo caso.