Informatica Teorica

Seconda prova in itinere – 30 Giugno 2008

Il tempo a disposizione è di 2 ore e 30 minuti

Esercizio 1 (punti 8/17-esimi)

Si formalizzi mediante formule del prim'ordine il seguente comportamento di un sistema di allarme a tempo continuo:

- **L'allarme si attiva e disattiva inserendo la chiave nell'apposito alloggiamento; mentre la disattivazione è immediata, l'attivazione avviene dopo 12 secondi dall'introduzione della chiave; un eventuale reinserimento della chiave durante questo periodo non ha effetto; l'estrazione della chiave dall'alloggiamento non ha alcun effetto e può essere trascurata.**
- **Ad allarme attivato l'ingresso di un corpo estraneo nel volume sotto controllo determina lo scatto dell'allarme dopo 12 secondi, a meno che durante tale intervallo non venga inserita la chiave nell'alloggiamento per disinserirlo.**

Si suggerisce di usare i seguenti predicati logici (dall'ovvio significato), tutti quanti aventi un parametro a valori reali: *attiva_allarme*(*t*), *disattiva_allarme*(*t*), *allarme_attivo*(*t*), *inserimento_chiave*(*t*), *rilevamento_corpo_estraneo*(*t*), *scatto_allarme*(*t*).

Esercizio 2 (punti 6/17-esimi)

Si dica, giustificando brevemente la risposta, se i seguenti problemi sono decidibili o semidecidibili:

1. **Stabilire se, data una generica macchina di Turing a nastro singolo e due configurazioni della medesima, le due configurazioni sono tra loro in relazione di transizione immediata.**
2. **Stabilire se, data una generica macchina di Turing a nastro singolo e due configurazioni della medesima, le due configurazioni sono tra loro in relazione di transizione non necessariamente immediata.**

Esercizio 3 (punti 6/17-esimi)

1. **Si supponga che una macchina di Turing deterministica simuli il comportamento di un automa a stati finiti nondeterministico nel seguente modo: ricevendo in ingresso una stringa, essa riproduce, una dopo l'altra, tutte le possibili esecuzioni che l'automa nondeterministico può svolgere su tale stringa. Si indichi la complessità minima (col criterio del caso pessimo) che tale simulazione richiede, sia per il tempo che per lo spazio, in funzione della lunghezza *n* della stringa in ingresso *x*.**
2. **Si supponga che una macchina di Turing deterministica abbia in ingresso la descrizione di un automa a stati finiti nondeterministico, e una stringa, da intendere come ingresso per l'automa, e scriva in uscita il valore 1 se la stringa appartiene al linguaggio accettato dall'automa, 0 altrimenti. Si valutino, giustificando brevemente la risposta:**
    a. **la complessità temporale e quella spaziale minime (col criterio del caso pessimo) come funzione del numero *m* degli stati dell'automa in ingresso;**
    b. **la complessità temporale e quella spaziale minime (sempre col criterio del caso pessimo) come funzione della lunghezza *n* della stringa in ingresso *x*.**

Soluzioni schematiche

Esercizio 1

***inserimento_chiave*, *attiva_allarme*, *disattiva_allarme*, *rilevamento_corpo_estraneo*, *scatto_allarme* sono definiti a priori come eventi, ossia istantanei, *allarme_attivo* è invece definito a priori come uno stato che ha quindi una durata. *UpToNow* applicato a uno stato è definito come in diversi esempi ed esercizi precedenti.**

Le formule seguenti esprimono i requisiti richiesti mediante la sintassi del calcolo dei predicati in due modi diversi.

$$\forall t \quad ((inserimento\_chiave(t) \wedge (\neg allarme\_attivo(t)) \wedge \rightarrow$$
$$((\forall t_1 (t \le t_1 < t+12) \rightarrow \neg allarme\_attivo(t_1)) \wedge attiva\_all(t+12))$$

$\wedge$

$$\forall t,k \quad (attiva\_all(t) \wedge ((\forall t_1 (t \le t_1 < t+k) \rightarrow \neg(inserimento\_chiave(t_1)) \rightarrow (\forall t_1 (t \le t_1 < t+k) \rightarrow allarme\_attivo(t_1))$$

$\wedge$

$$\forall t \quad ((inserimento\_chiave(t) \wedge (UpToNow(allarme\_attivo(t)) \rightarrow disattiva\_all(t))$$

$\wedge$

$$\forall t,k \quad (disattiva\_all(t) \wedge ((\forall t_1 (t \le t_1 < t+k) \rightarrow \neg(inserimento\_chiave(t_1)) \rightarrow (\forall t_1 (t \le t_1 < t+k) \rightarrow \neg allarme\_attivo(t_1))$$

$\wedge$

$$\forall t \quad allarme\_attivo(t) \wedge rilevamento\_corpo\_estraneo(t) \wedge (\forall t_1 (t \le t_1 < t+12) \rightarrow allarme\_attivo(t_1)) \rightarrow scatto\_allarme(t+12)$$

$\wedge$

$$\forall t \quad \neg allarme\_attivo(t) \rightarrow \neg scatto\_allarme(t)$$

Soluzione alternativa:

$$\forall t \begin{pmatrix} attiva\_allarme(t) \leftrightarrow \\ \quad inserimento\_chiave(t-12) \wedge \neg allarme\_attivo(t-12) \wedge \\ \quad \forall t_1 (t-12 < t_1 < t \rightarrow \neg attiva\_allarme(t_1)) \end{pmatrix}$$

$\wedge$

$$\forall t \left(disattiva\_allarme(t) \leftrightarrow inserimento\_chiave(t) \wedge allarme\_attivo(t) \right)$$

$\wedge$

$$\forall t \begin{pmatrix} allarme\_attivo(t) \leftrightarrow \\ \quad \exists t_1 \begin{pmatrix} t_1 < t \wedge attiva\_allarme(t_1) \wedge \\ \forall t_2 (t_1 < t_2 < t \rightarrow \neg disattiva\_allarme(t_2)) \end{pmatrix} \end{pmatrix}$$

$\wedge$

$$\forall t \begin{pmatrix} scatto\_allarme(t) \leftrightarrow \\ \quad rilevamento\_corpo\_estraneo(t-12) \wedge allarme\_attivo(t-12) \wedge \\ \quad \forall t_1 (t-12 < t_1 < t \rightarrow \neg inserimento\_chiave(t_1)) \end{pmatrix}$$

Esercizio 2

Quesito 1

Il problema è sicuramente decidibile: è infatti immediato costruire un semplice algoritmo che scandisca la configurazione iniziale e determini il simbolo in lettura e lo stato della macchina, individui la mossa definita dalla funzione δ della macchina e verifichi se la seconda configurazione coincide con quella che si otterrebbe applicando la δ.

Quesito 2

Questo problema è invece non decidibile. Infatti il classico problema dell'HALT è un caso particolare del problema posto: stabilire cioè se da una configurazione iniziale si può giungere a una configurazione di HALT. NB: per la precisione, visto che in generale vi possono essere diverse di configurazioni di HALT, per ricondurre il problema dell'HALT esattamente al problema in questione è necessario modificare la macchina in modo tale che essa abbia un'unica configurazione di HALT (ad esempio, nastro vuoto e stato q_F.)

Il problema è invece semidecidibile in quanto è sufficiente "far girare" la macchina a partire dalla prima configurazione: se la seconda è raggiungibile, prima o poi la si raggiunge e si risolve il problema. Non vale ovviamente il viceversa.

Esercizio 3

Quesito 1

Il meccanismo di simulazione della MT è quello classico di visita dell'albero delle computazioni; quindi la complessità temporale è nel caso pessimo $\Theta(2^n)$ (per semplicità assumiamo che l'albero sia binario). Se la macchina costruisce l'intero albero delle computazioni anche la complessità spaziale è dello stesso ordine. Tuttavia la macchina potrebbe limitarsi a memorizzare solo l'ultimo cammino visitato, ad esempio, in preordine sinistro; in tal modo potrebbe generarli ed esaminarli tutti in sequenza (ad esempio:

S, S, S, S, S -→ S,S,S,S,D → S, S, S, D, S → S, S, S, D, D → S, S, D, S, S ….)

senza costruire l'intero albero. In tal caso la complessità spaziale risulta $\Theta(n)$.

Quesito 2

Consideriamo una macchina di Turing che costruisce l'automa deterministico equivalente a quello non deterministico di partenza, e quindi effettua il riconoscimento della stringa simulando direttamente l'automa deterministico.

a.

E' noto che nel caso pessimo l'automa deterministico, equivalente a quello in input di m stati, ha $\Theta(2^m)$ stati e quindi la complessità spaziale della macchina è dominata da questo fattore. Allo stesso modo, la complessità temporale è dominata dalla costruzione di tale automa.

b.

Una volta costruito l'automa deterministico la MT può semplicemente simulare quest'ultimo con la stessa sua complessità, ossia $\Theta(n)$ per la complessità temporale e $\Theta(1)$ per quella spaziale, tenendo conto ovviamente che la memoria finita costituita dagli $\Theta(2^m)$ stati non è funzione della lunghezza della stringa di ingresso *x*.

Soluzione alternativa del Quesito 2:

Considerando i quesiti a e b come *indipendenti*, possiamo scegliere un algoritmo diverso in ciascuno dei due casi, puntando unicamente a minimizzare la complessità rispetto al parametri considerato nel punto in questione.

Per il punto b la soluzione basata sulla determinizzazione suggerita nella soluzione precedente rimane la migliore dal punto di vista della complessità ottenibile.

Per il punto a, invece, possiamo utilizzare di nuovo l'algoritmo di simulazione del punto 1. La complessità spaziale è quella necessaria a memorizzare l'ultimo cammino visitato; essendo un cammino lungo n ed essendo necessari log m bit per memorizzare (sinteticamente in binario) l'informazione su uno degli m stati, abbiamo una complessità spaziale di $\Theta(\log(m))$ visto che n non è funzione di m. Similmente, la complessità temporale è quella necessaria ad esplorare l'albero delle 2^n computazioni, dove ogni passo costa l'aggiornamento dello stato corrente, ossia log(m). Quindi anche la complessità temporale è $\Theta(\log(m))$. Ovviamente questa valutazione di complessità è formalmente corretta ma irrealistica perchè ignora completamente la dimensione di parte dell'input (e i suoi effetti sulla complessità).