Informatica Teorica

Appello d'esame - 3 Settembre 2010
Tempo a disposizione: 2h

Esercizio 1 (8 punti)
Si definisca un automa in grado di riconoscere il seguente linguaggio:

$$L=\{\mathit{bin}(i)\$[\mathit{bin}(i+1)]^R | i>=0\}$$

dove *bin(k)* indica la rappresentazione in binario (con il minor numero di bit possibile) del numero naturale k e l'apice R il riflesso, cioè nelle parole che fanno parte di L il primo numero (bin(i)) è scritto con la cifra più significativa a sinistra, mentre il secondo (bin(i+1)) con la cifra più significativa a destra.
L'automa deve essere a potenza minima tra quelli che riconoscono il linguaggio L.

Esempi di stringhe che appartengono al linguaggio: 1111$00001, 1000$1001, 100101$011001.
Esempi di stringhe che non appartengono al linguaggio: 001$010, 100$111, 0011$0010.

Esercizio 2 (9 punti)
Si introduca il predicato *nextS(t1,t2)* che è vero se e solo se

- *t2 > t1*
- all'istante *t1* viene emesso il segnale S
- la volta successiva a *t1* che il segnale S viene emesso è all'istante *t2*.

Utilizzando il solo predicato *nextS* (ed eventualmente relazioni e operazioni aritmetiche quali >, <, =, +, -, ecc.) si formalizzino mediante formule logiche le seguenti proprietà (tra di loro indipendenti):

1) Il segnale S viene emesso al massimo una volta sola.
2) Il segnale S viene emesso la seconda volta all'istante 10.
3) Il segnale S viene emesso un numero finito di volte, maggiore o uguale a 2.
4) Il segnale S viene emesso un'infinità di volte.

Il tempo è da intendersi discreto.

Esercizio 3 (8 punti)
Si considerino gli insiemi:

$S = \{i \mid \forall x\ (f_i(x) = \perp \vee f_i(x) \leq x)\};$

$T = \{i \mid \exists x\ \exists k\ (k > 0 \wedge f_i(x) = x+k)\}.$

Si dica se i seguenti insiemi sono decidibili, semidecidibili o altro:
1) S
2) T
3) $S \cap T$.

Esercizio 4 (9 punti)
Si consideri il linguaggio L fatto di stringhe di parentesi tonde ben parentetizzate terminate dal simbolo *$*.
Esempi di stringhe appartenenti ad L sono: ()()$, (())$, ((())()(()))()(())$.
Esempi di stringhe *non* appartenenti ad L sono: ())$,)()($, (()))($.

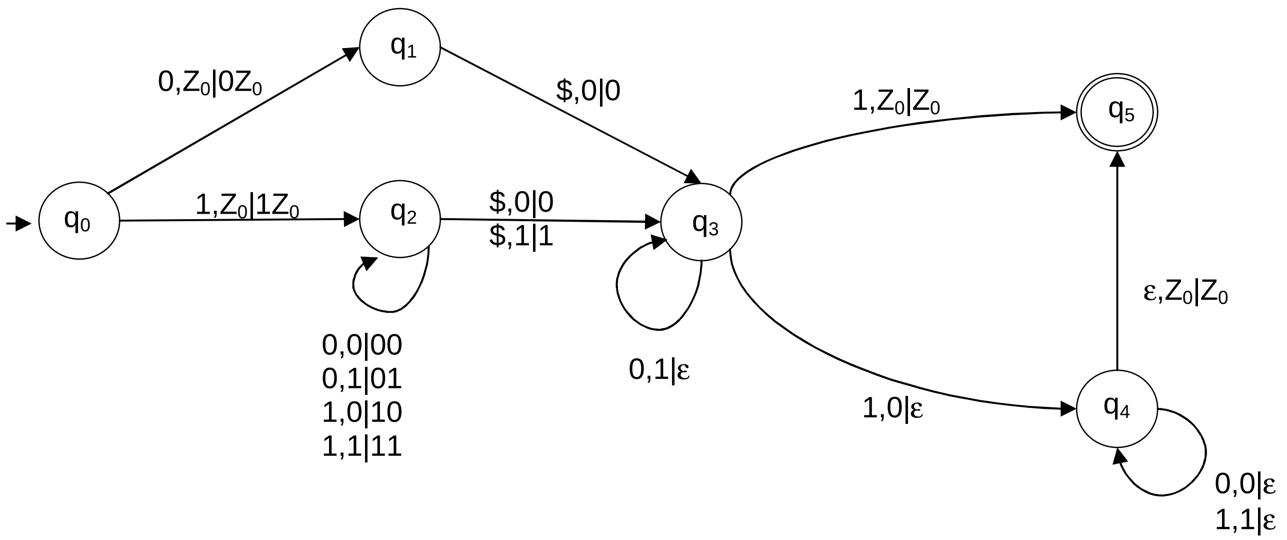
1) Si delinei la più semplice Macchina di Turing a k nastri che accetta il linguaggio L, e se ne valutino le complessità spaziale e temporale in funzione della lunghezza *n* della stringa in input.

   1a) Come cambiano, se cambiano, le complessità se si usa una Macchina di Turing *a nastro singolo* invece di una a k nastri?

2) Si delinei la più semplice macchina RAM che accetta il linguaggio L, e se ne valutino le complessità temporale e spaziale in funzione della lunghezza *n* della stringa di input, usando il criterio di *costo logaritmico*.

Soluzioni

Esercizio 1
Per riconoscere L è sufficiente il seguente AP deterministico (in cui, per rappresentare la pila, si adotta la convenzione sinistra/alto, destra/basso):



Esercizio 2

1) ¬∃t1,t2 (nextS(t1,t2))
2) ∃t1 (nextS(t1,10) ∧ ¬∃t' (nextS(t',t1))
3) ∃t1,t2 (nextS(t1,t2) ∧ ¬∃t'(nextS(t2,t'))
4) ∃t1,t2 (nextS(t1,t2)) ∧ ∀t3,t4(nextS(t3,t4) → ∃t'(nextS(t4,t')))

Esercizio 3

1) Indecidibile, non semidecidibile (si vede facilmente che il suo complemento è semidecidibile, con la solita tecnica di esecuzione un passo alla volta di $f_i(x)$ a partire da x=0).
2) Semidecidibile (stesso approccio).
3) Decidibile, visto che è l'insieme vuoto.

Esercizio 4
A 1-tape Turing machine can scan the input and keep a counter (in unary notation, for instance storing '*' marks in its memory tape) of the currently reached nesting level, increasing the counter when reading a '(' symbol, and decreasing it when reading a ')' symbol. It outputs NO if the counter becomes negative or it outputs YES if it reaches the end of the string with a null counter. The time and space complexity are therefore both Θ(n).
A single tape Turing machine can use a blank portion of its (unique) tape to store the counter in unary form. The space complexity is unchanged, but the time complexity is increased: in the worst case the TM must make Θ(n) moves to update the counter for every input symbol, so the time complexity is $\Theta(n^2)$.
A RAM program can basically adopt the same algorithm as the 1-tape TM, but storing the counter, in binary form, in a single memory cell. Using the logarithmic cost criterion, the worst case is that of a string of length *n*=2·*k**1 composed of *k* '(' symbols followed by *k* ')' symbols and the '$' sign. For this string the space complexity (size of the cell storing the counter) is Θ(log(*n*)); for the time complexity the dominating factor is related with the increase and decrease operations on the counter value. These are both proportional to $\sum_{i=1}^{n} \log(i) = \log(n!) = n \cdot \log(n)$, hence the time complexity is Θ(n·log(n)).