Informatica Teorica

Appello d'esame - 13 Luglio 2010
Tempo a disposizione: 2h

Esercizio 1 (14 punti)
Si consideri il seguente linguaggio costruito sull'alfabeto I={a,b}. Le stringhe del linguaggio sono tali che ogni simbolo *a* è seguito da almeno *n*=3 simboli, ed il terzo simbolo seguente è una *b*.

1. Si scriva un automa a stati finiti deterministico che accetta il linguaggio descritto sopra.
2. Si descriva come tale automa a stati finiti deterministico potrebbe essere modificato per trattare il caso in cui *n*=4, o in generale un qualunque numero maggiore di 3. In particolar modo, si dica precisamente quanti stati ha l'automa a stati finiti generalizzato in dipendenza dal valore di *n*.
3. Si consideri ora la nozione di automa a stati finiti *universalmente nondeterministico*. Questo è definito come un usuale automa a stati finiti nondeterministico, salvo per il fatto che una stringa *x* è accettata se e solo se *tutte* le possibili computazioni nondeterministiche accettano *x*, cioè terminano in uno stato finale. Formalmente, il linguaggio accettato *L* è definito, invece che dalla formula $x \in L \leftrightarrow (\delta^*(q_o,x) \cap F \neq \emptyset)$ come negli abituali automi a stati finiti nondeterministici, dalla seguente formula: $x \in L \leftrightarrow (\delta^*(q_o,x) \subseteq F)$.
   Si definisca un automa a stati finiti *universalmente nondeterministico* che accetta il linguaggio descritto sopra con *n*=3. Si sfrutti questa forma di nondeterminismo per definire un automa con il minor numero di stati possibile.
4. Si descriva come l'automa stati finiti universalmente nondeterministico potrebbe essere modificato per trattare il caso in cui *n*=4, o in generale un qualunque numero maggiore di 3. In particolar modo, si dica precisamente quanti stati ha tale automa a stati finiti universalmente nondeterministico generalizzato in dipendenza del valore di *n*.

Esercizio 2 (10 punti)
Sia *f(y,x)* la funzione che restituisce il valore computato dalla *y*-esima Macchina di Turing quando questa in ingresso ha il valore *x*.

Per ognuna della seguenti formule logiche dire, spiegando in modo adeguato il perché, se esiste una Macchina di Turing che è in grado di calcolare il valore di verità delle formule al variare del valore delle variabili che appaiono in esse.

1) $\exists y,z\ \forall x\ (f(y,x) = f(z, x^2+1))$

2) $\forall x\ (f(y,x) = f(z,x+1))$

3) $\exists z\ (z \neq y \wedge \forall x(f(y,x) = f(z,x)))$

4) $\forall x(f(y,x) = f(z,x))$
$\wedge$
$\forall x(\ (f(y,x) = 1 \wedge \forall w(f(x,w) \neq \perp))$
$\vee$
$(f(y,x) = 0 \wedge \exists w(f(x,w) = \perp))\ )$

(continua dietro)

Esercizio 3 (10 punti)
Si consideri una macchina di Turing che accetta il seguente linguaggio: array bidimensionali (quadrati o rettangolari) di zeri e uni in cui esattamente una riga e una colonna sono composti di caratteri 1, mentre il resto è fatto da caratteri 0. La figura viene fornita in ingresso alla macchina riga per riga, con il carattere $ come separatore di riga.
Dato un ingresso di questo genere, la macchina:

- controlla che le righe siano tutte della stessa lunghezza;
- controlla che ci siano esattamente una riga e una colonna composte da soli 1.

Se il dato in ingresso soddisfa le condizioni, la macchina emette in uscita il numero complessivo di 1 in esso presenti, altrimenti si ferma in uno stato non finale senza emettere nulla.

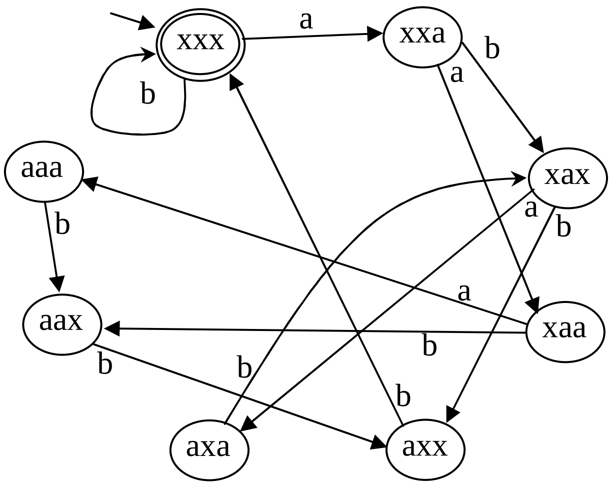
Si descriva in maniera sufficientemente accurata ma ad alto livello il funzionamento di una MT che effettui la precedente traduzione, minimizzandone il più possibile la complessità spaziale. Se ne valutino inoltre sia la complessità temporale che quella spaziale.

NB: non è necessario fornire il diagramma delle transizioni della macchina, basta per esempio una descrizione in pseudocodice, includendo comunque la valutazione esplicita della complessità spaziale e temporale.

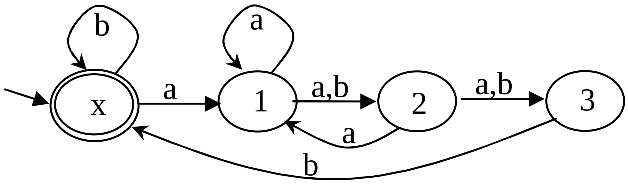
Soluzioni

Esercizio 1

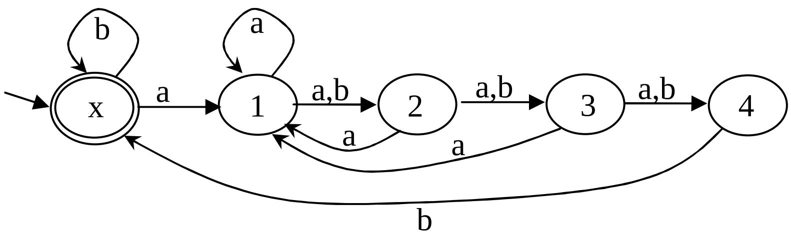
1. In the proposed solution, the a's appearing in the states represent symbols occurred in the string analyzed so far that still must be matched by a corresponding *b*.



2. To deal with the general case of *n* symbols, the automaton should have 2^n states and keep memory of all the *a*'s possibly occurring in last *n* symbols of the string analyzed so far.
3. With universal nondeterminism *n+1* states suffice. For every *a* occurring in the string the automaton must have a computation where a *b* occurs after *n* positions, starting from that pint. State labeled *k* (k=1, 2, ... , n) means: the *k*-th last symbol was an *a*. State labeled *x* means: any previous a symbol at distance >*n* has been matched by a successive *b*. Here is the solution for n=3...



4. ... and for n=4



Esercizio 2

1)
La MT esiste, in quanto la formula è chiusa, e quindi o è sempre vera o è sempre falsa, indipendentemente dai valori di *y*, *z*, *x*; di conseguenza, la MT in questione o è quella che ritorna sempre 1 (vero) o sempre 0 (falso).

2)
In questo caso la formula è vera per tutte e sole le coppie *y*,*z* di (indici di) MT tali che f_z corrisponde a f_y scalata di 1. Se esistesse una MT che calcola il valore di verità di questa formula al variare di *y* e *z*, allora sarebbe decidibile il problema di stabilire se, data una qualunque coppia *y*, *z* di (indici di) MT, esse sono tali

che f_z corrisponde a f_y scalata di 1. Tale problema però non è decidibile (è possibile farlo vedere fissando un particolare valore di *y*, ed applicando il teorema di Rice), quindi la MT in questione non esiste.

3)
La formula dice che, preso un generico indice di MT *y*, è possibile trovare una MT di indice *z* diversa dalla prima che calcola la stessa funzione. Questo è sempre vero (data una qualunque MT, è banale costruirne una diversa che calcola la stessa funzione, per esempio introducendo uno stato iniziale fittizio nel quale la nuova MT non fa nulla), quindi una MT che calcola la funzione desiderata è semplicemente una che come risultato dà sempre 1 (vero).

4)
In questo caso la formula è vera per tutte le coppie di (indici di) MT *y*, *z* che calcolano la stessa funzione, ma tali che la funzione calcolata da *y* (e quindi da *z*) è quella che, preso un qualunque indice di MT *x*, ritorna 1 se f_x è una funzione totale, 0 altrimenti. Una tale funzione però non è calcolabile, quindi l'insieme di coppie *y*,*z* suddetto è vuoto, e la MT richiesta è semplicemente quella che dà sempre come risultato falso (0).

Esercizio 3

Sia:
- m: numero di righe
- n: numero di colonne
- LR: contatore binario per lunghezza di ogni riga (lg n)
- IC: contatore binario per indice di colonna (lg n)
- NR: contatore binario per la somma di 1 (lg (n + m))
- X: contatore binario temporaneo (lg n)

in cui la complessità spaziale è indicata tra parentesi

Gli stati dell'organo di controllo servono ad assicurarsi che ad ogni riga si trova un unico 1, e che una unica riga è fatta di 1.

La MT si comporta nel seguente modo (le complessità dei singoli passi sono indicate tra parentesi):

```
LR := 0
leggi fino a $, incrementando LR ad ogni passo - quando trovi 1, copia LR in IC. (n lg n)
NR := 0
per ogni riga successiva: (m)
        incrementa NR (lg (n + m), eseguito m volte, una per ogni riga)
        X := 0 (lg n)
        se riga normale: leggi la riga e incrementa X ad ogni passo, fino all'1;
                se X = IC continua e alla fine controlla che X = LR    (n lg n, ripetuta m-1 volte)
        se riga di 1: leggi la riga e incrementa X e NR ad ogni passo, fino a $;
                poi controlla che X = LR  (n lg (n + m), eseguita solo per la riga fatta da 1)
restituisci NR in uscita (lg (n + m))
```

complessità spaziale S = Θ(lg (n + m))
complessità temporale T = Θ(m n lg n + (m + n) lg (m + n))