Algoritmi e Principi dell'Informatica
Seconda Prova in Itinere

14 Febbraio 2014

Avvisi importanti
Il tempo a disposizione è di 1 ora e 30 minuti.
Se non verranno risolti in maniera soddisfacente gli esercizi 1, 2, (ossia ottenendo almeno 6 punti in totale tra i due) non si procederà alla correzione degli altri esercizi; l'esercizio facoltativo 4, sarà valutato solo se si saranno ottenuti almeno 13 punti nei primi 3.

Esercizio 1 (punti 4/15-esimi)

Si consideri un albero binario in cui ogni nodo o sia una foglia o abbia entrambi i figli. Quali sono l'altezza minima e massima di un tale albero con n nodi?
NB: si richiede il *valore preciso della funzione di n*, non basta l'ordine di grandezza!

Esercizio 2 (punti 3/15-esimi)

E' noto che esistono linguaggi riconoscibili in tempo lineare da automi a pila deterministici e da macchine di Turing a k nastri ma non da macchine di Turing a nastro singolo.
Esistono anche linguaggi regolari non riconoscibili in tempo lineare da macchine di Turing a nastro singolo? Giustificare brevemente la risposta.

Esercizio 3 (punti 8/15-esimi)

Si descriva un algoritmo che, dato un BST (Binary Search Tree) stabilisca se esso *possa* essere colorato in modo tale da diventare un albero rosso-nero (RB).
NB: non si chiede di costruire un algoritmo di colorazione, ma solo di stabilire se ciò sia possibile.
Si ricordi inoltre che per convenzione in un albero RB tutte le foglie sono NIL, T.NIL per la precisione, e nere.
Si valuti la complessità temporale dell'algoritmo.

Esercizio 4, facoltativo (punti 3/15)

Si formalizzi la seguente definizione informale di complessità spaziale di una macchina di Turing a nastro bidimensionale:

> Il nastro della macchina è costituito dal primo quadrante di un piano cartesiano a coordinate intere. La complessità spaziale è data dal ***numero di celle del nastro visitate dalla testina*** della macchina durante il suo funzionamento; al solito, essa è espressa come funzione della lunghezza della stringa (lineare) di ingresso.

Tracce di soluzioni

Esercizio 1
L'altezza massima si ha quando ogni nodo interno ha, ad esempio, un foglia come figlio sinistro, tranne l'ultimo i cui figli sono entrambi foglia. In tal caso, detti h l'altezza dell'albero, (definita come il cammino di lunghezza massima, intesa come il numero di archi del cammino, tra la radice e le foglie) e n il numero di nodi, si ha
$n = 2.h + 1$; quindi
$h = \frac{n-1}{2}$
L'altezza minima si ha invece quando l'albero è bilanciato e completo, ossia
$n = \sum_{i=0}^{h} 2^i = 2^{h+1} - 1$; quindi
$h = log_2(n+1) - 1$.

Esercizio 2
NO: una MT a nastro singolo, una volta posizionata all'inizio delle stringa di ingresso può comportarsi esattamente come un automa a stati finiti, usando il proprio organo di controllo. Siccome un automa a stati finiti esegue sempre uno spostamento a destra della testina di lettura per ogni mossa, la stessa cosa può fare la MT senza dover utilizzare altre porzioni del nastro.

Esercizio 3
Una condizione necessaria e sufficiente perché un BST possa essere colorato in modo da diventare un albero rosso-nero è che, nell'intero albero e in ogni suo sottoalbero il rapporto tra le lunghezze minime e massime dei cammini che dalla radice raggiungono le foglie non superi mai 2. Infatti in tal caso sarà sempre possibile colorare i nodi in modo da evitare due nodi rossi consecutivi sullo stesso cammino e l'altezza di ogni sottoalbero sarà comunque $\Theta(\log(n))$.
Un semplice algoritmo ricorsivo potrebbe quindi visitare l'albero, ad esempio in preordine sinistro, e calcolare, per ogni nodo, la lunghezza dei cammini minimo e massimo da esso alle foglie (min = min(leftSon.min, rightSon.min) +1; max = max(leftSon.max, rightSon.max) +1; min e max dei nodi foglia sono ovviamente 0). Non appena max/min >2 l'algoritmo ritorna FALSE; se giunge fino alla radice con max/min <= 2, ritorna TRUE. Si noti però che, rispetto a un normale BST in un albero rosso-nero le foglie sono tutte NIL (T.NIL per la precisione) e hanno convenzionalmente altezza (nera) = 0; ogni nodo interno ha quindi entrambi i figli.
L'algoritmo ha evidentemente complessità $\Theta(n)$.

Esercizio 4
Una possible definizione formale della complessità spaziale richiesta può essere fornita nel modo seguente:
In primo luogo occorre formalizzare il concetto di configurazione:
$c = \langle q, x, \text{contenuto}: \mathcal{N} \times \mathcal{N} \rightarrow A, \langle i,j \rangle, h \rangle$
dove q rappresenta lo stato dell'organo di controllo, x la stringa in ingresso, contenuto indica il simbolo contenuto in ogni cella del nastro (con l'usuale vincolo che solo un numero finito di celle sia diverso da blank), $\langle i,j \rangle$ indica la posizione della testina del nastro di memoria e h quella del nastro di ingresso. La transizione tra configurazioni è poi definita nel modo tradizionale.

La complessità spaziale di M è allora espressa dalla funzione:
$S_M(n) = \max_{|x| = n|} |\{\langle i,j \rangle \mid \exists c\ \langle i,j \rangle \in c\ \wedge$
$c_0 = \langle q_0, x, [\text{contenuto}(0, 0) = Z_0, \text{contenuto}(i, j) = \text{blank}\ \forall \langle i,j \rangle \neq \langle 0,0 \rangle], \langle 0,0 \rangle, 0] \mid\!\!\text{---}^* c\}|$

Ossia la massima cardinalità, rispetto a tutte le stringhe di lunghezza n, dell'insieme delle coordinate delle posizioni della testina di memoria raggiunte durante il funzionamento della macchina a partire dalla configurazione iniziale.