Algoritmi e Principi dell'Informatica
Soluzioni al Tema d'esame

12 febbraio 2024

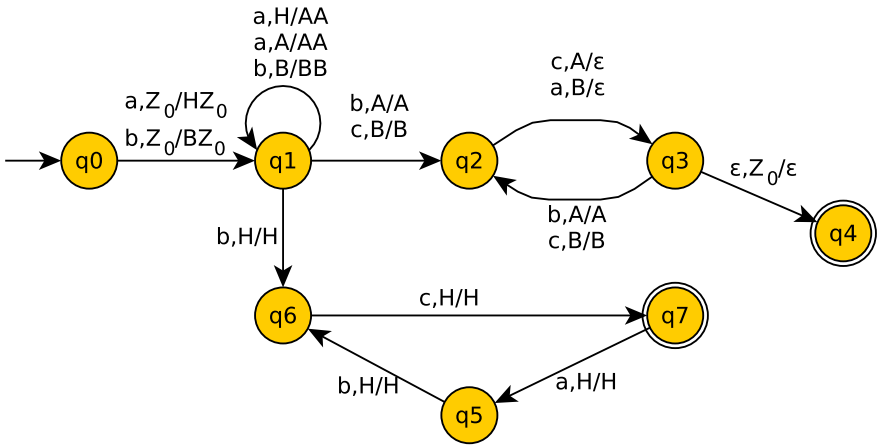
Informatica teorica

Esercizio 1 (8 punti)

Si consideri il linguaggio $L = \{a^n(bc)^n \mid n \geq 2\} \cup \{b^n(ca)^n \mid n \geq 1\} \cup (abc)^+$. Si dica a che classe di linguaggi appartiene, motivando la risposta, e si costruisca un automa a potenza minima che lo riconosce.

Soluzione

Il linguaggio è chiaramente non regolare, richiedendo capacità di conteggio illimitate. Esso si può riconoscere con il seguente automa a pila deterministico, quindi si tratta di un linguaggio deterministico non contestuale.



Esercizio 2 (8 punti)

Si consideri un sistema, chiamato BeeS, che accetta sequenze di comandi, in cui ogni comando è rappresentato da un opportuno simbolo di un alfabeto. Il comportamento di tale sistema (cioè l'insieme delle sequenze ammissibili di comandi) è descritto dall'automa a stati finiti A. Sia S un insieme (finito) fissato di sequenze di comandi, e G un algoritmo (anch'esso fissato) che, dato in input l'automa A e un insieme finito di sequenze di comandi, restituisce un automa a stati finiti che viene appreso a partire dalle sequenze in input (l'algoritmo termina sempre, per qualunque insieme di sequenze fornito in input).

1. È decidibile il problema di stabilire se, fornendogli in input l'automa A e l'insieme S di sequenze, l'algoritmo G restituisce un automa equivalente ad A?
2. È decidibile il problema di stabilire se, preso un qualunque automa a stati finiti A' e un insieme di sequenze S', se all'algoritmo G vengono forniti in input l'automa A' e l'insieme S', esso restituisce un automa equivalente ad A'?
3. È decidibile il problema di stabilire se, preso un qualunque algoritmo G', tale algoritmo, per ogni automa a stati finiti A' e insieme di sequenze S', restituisce un automa che è equivalente ad A'?

Soluzione

1. A, S, G sono tutti fissati, quindi il problema è in forma chiusa, ed è banalmente decidibile.

2. Il problema è ancora decidibile, perché è sufficiente, preso un qualunque automa A' e insieme di sequenze S', produrre l'automa A'' mediante G (che è garantito terminare), e controllare se A'' e A' sono equivalenti, che è un problema notoriamente decidibile

3. In questo caso il problema è indecidibile, perché corrisponde al problema di determinare se G' calcola la funzione che, dati in input un automa e una sequenza di eventi, restituisce un automa equivalente all'automa in input (che è una funzione calcolabile, basterebbe banalmente restituire l'automa in input, ma chiaramente ci sono funzioni calcolabili diverse da essa).

Algoritmi e Principi dell'Informatica
Soluzioni al Tema d'esame

12 febbraio 2024

Algoritmi e strutture dati

Esercizio 3 (8 punti)

Si consideri il linguaggio $L = \{a^n(bc)^n \mid n \geq 2\} \cup \{b^n(ca)^n \mid n \geq 1\} \cup (abc)^+$. Si descrivano con sufficiente precisione una macchina di Turing a nastro singolo, una MT ad un nastro e una RAM che accettino L e se ne valutino le complessità spaziali e temporali, utilizzando il criterio di costo più opportuno.

Soluzione

Per prima cosa notiamo come tutte le macchine richieste possano controllare in tempo (e spazio) costante i primi due caratteri della stringa per capire in quale caso si trovi: aa per il primo, bb o bc per il secondo, ab per il terzo.

1. **MT a nastro singolo.** Nel primo caso la macchina effettua il solito conteggio con un ciclo di scansioni della stringa in ingresso, riscrivendo come X ad ogni scansione a e bc, trovati immediatamente dopo le X. Il ciclo viene inizializzato con la riscrittura in X dei primi a e bc. Il caso 2 è analogo, però con b e ca. Il caso 3 è una semplice scansione lineare di controllo per abc ripetuto.

   Sia n la lunghezza della stringa in ingresso. La complessità temporale è $\mathcal{O}(n^2)$ per i primi 2 casi e $\mathcal{O}(n)$ per il terzo, quindi $T(n) = \mathcal{O}(n^2)$. Da un punto di vista spaziale, la macchina non necessita di altra memoria per il suo funzionamento, dunque $S(n) = \mathcal{O}(n)$.

2. **MT a 1 nastro.** In questo caso possiamo sfruttare il nastro di memoria per i conteggi dei primi 2 casi, evitando scansioni ripetute della stringa. Ricordando che, con opportuni accorgimenti, il costo complessivo di n operazioni di incremento/decremento di un contatore binario è $\mathcal{O}(n)$, abbiamo $T(n) = \mathcal{O}(n)$ e $S(n) = \mathcal{O}(\log n)$.

3. **RAM.** L'unica memoria significativa per il funzionamento della macchina è il contatore considerato nel punto precedente. Per questo genere di linguaggio il criterio di costo costante è sufficiente, visto che anche con il criterio logaritmico non si uscirebbe dalla classe P, quindi otteniamo $T(n) = \mathcal{O}(n)$ e $S(n) = \mathcal{O}(1)$.

Esercizio 4 (8 punti)

Si consideri una collezione che ha un numero di elementi n_0 al momento della sua inizializzazione. Ad ogni istante di tempo t viene aggiunto alla collezione un certo quantitativo di elementi; tale quantitativo è funzione del numero di elementi n_{t-1} all'istante precedente, ossia $n_t = n_{t-1} + f(n_{t-1})$. Si considerino i casi:

1. $f(x) = \frac{x}{k}$, dove k è una costante intera positiva.
2. $f(x) = k$, dove k è una costante intera positiva.

Si indichi una possibile struttura dati che consenta la ricerca di un elemento all'interno della collezione al tempo t con complessità temporale $\mathcal{O}(\log(n_t))$. Si analizzi la complessità temporale dell'inserimento complessivo degli $f(n_{t-1})$ elementi aggiunti al tempo t e la si esprima in funzione di n_0 e di t per i casi 1 e 2 suesposti.

Soluzione

Allo scopo di garantire la ricerca in tempo logaritmico è possibile utilizzare un albero di ricerca con profondità limitata (e.g. alberi rosso-neri). Ad ogni aggiunta alla collezione è sufficiente utilizzare le procedure di inserimento e bilanciamento.

1. La dimensione della collezione di dati può essere espressa in funzione di t come $n_t = n_0 \left(\frac{k+1}{k}\right)^t$. Posto $c = \frac{k+1}{k}$, inserire un elemento al tempo t ha quindi costo

$$\begin{gathered}\mathcal{O}\left(\frac{n_{t-1}}{k}\log(n_{t-1})\right) = \\ \mathcal{O}\left(\frac{n_0 c^{t-1}}{k}\log\left(n_0 c^{t-1}\right)\right) = \\ \mathcal{O}\left((n_0 c^{t-1}) \cdot (\log(n_0) + t\log(c))\right) = \\ \mathcal{O}\left((n_0 c^{t-1}) \cdot (\log(n_0) + t)\right).\end{gathered}$$

2. La dimensione della collezione di dati può essere espressa in funzione di t come $n_t = n_0 + kt$, da cui ogni inserimento ha costo $\mathcal{O}(\log(n_0 + kt)k) = \mathcal{O}(\log(n_0 + kt))$.