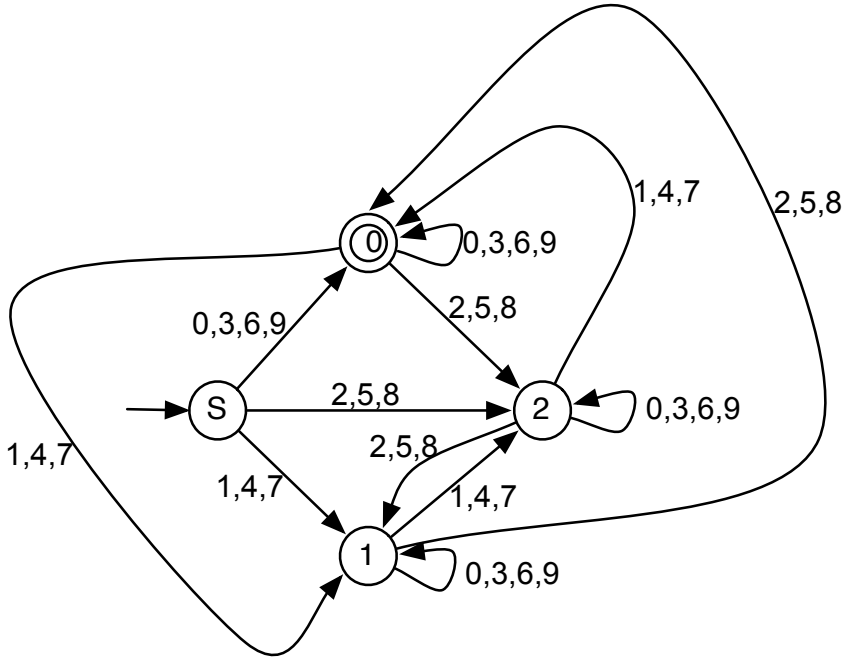
Soluzioni

Esercizio 1

Basta un automa a stati finiti che tenga conto della somma delle cifre modulo 3 (più uno stato iniziale per evitare di accettare la stringa vuota).



Esercizio 2

Punto 1.a

Si tratta di una domanda con risposta sì/no che non dipende da alcun input, pertanto il problema è decidibile.

Punto 1.b

Anche in questo caso la risposta non dipende da alcun input, pertanto il problema è decidibile.

Punto 2

Non è possibile scrivere il programma richiesto. Se per assurdo fosse possibile, allora si potrebbe anche risolvere il problema dell'arresto, che è invece notoriamente indecidibile.

Per vederlo, basta procedere come segue. Si consideri un generico programma P e si assuma, senza perdita di generalità, che `x` non sia una variabile usata da P. Si costruisca una funzione `g` nel modo seguente:

```
int g(int x) {
  // qua viene inserito il codice di P
  x = -1;
}
```

Allora, per la chiamata `g(3)` risulta costantemente `x` > 0 durante l'intera esecuzione di `g` se e solo se P non termina la sua esecuzione.

Esercizio 3

Punto 1

La formalizzazione è identica agli automi a stati finiti nondeterministici (a parte il simbolo #). Cambia solo la definizione di linguaggio accettato:

$$L(A) = \{x \mid \delta^*(q_0, x\#x^R) \cap F \neq \emptyset\}.$$

Punto 2

Il potere espressivo è equivalente a quello degli automi a stati finiti.

E' immediato mostrare che l'automa a stati finiti a due passate (2PFSA) riesce a fare tutto quello che può fare un NFSA: dato un NSFA, si costruisce un 2PFSA equivalente utilizzando lo stesso diagramma degli stati, con in più delle transizioni che

- alla lettura del carattere # si spostano in un nuovo stato di accettazione S_1 se la lettura è avvenuta in uno stato di accettazione, o di non accettazione S_2 altrimenti
- una volta entrati in S_1 o in S_2 vi si rimane indipendentemente da quello che viene letto in ingresso.

Per mostrare che un NSFA può fare tutto quello che fa un 2PFSA A si può procedere come segue.

Si costruisce un NSFA A_1 che ha lo stesso diagramma di stato di A, ma in cui vengono considerati finali tutti e soli gli stati a partire dai quali c'è un arco uscente etichettato con #; inoltre tutti questi archi vengono eliminati dal diagramma di A_1. L'idea è che A_1 riconosce le stringhe che A legge fino al separatore # escluso.

Si costruisce poi un NSFA A_2 che ha lo stesso diagramma di stato di A, ma in cui vengono considerati finali tutti e soli gli stati con un arco entrante etichettato con #; inoltre tutti questi archi vengono eliminati dal diagramma di A_2, il senso di tutte le altre frecce viene invertito, e si aggiunge un nuovo stato iniziale che con una ϵ-mossa va in tutti gli stati che erano finali per A. L'idea è che A_2 riconosce le inverse delle stringhe che A legge da dopo il separatore # fino alla fine.

Le stringhe riconosciute dal 2PFSA A sono quelle in cui la stringa viene letta in avanti fino al separatore e poi al contrario, da dopo il separatore alla fine.

Queste due condizioni di lettura sono date dai due NFSA A_1 e A_2, pertanto un NFSA equivalente ad A è dato dall'intersezione di A_1 e A_2.