Algoritmi e Principi dell'Informatica
9 febbraio 2017

Nome:_____

Matricola:_____

Firma:_____

Esercizio 1 (8 punti)

Si considerino i due seguenti problemi:

P1 = determinare se una generica macchina di Turing accetta al più 100 stringhe diverse.

P2 = determinare se una generica macchina di Turing accetta più di 100 stringhe diverse.

Rispondere, fornendo opportune motivazioni, ai seguenti quesiti:

- P1 è decidibile?

- P2 è decidibile?

- P1 è semidecidibile?

- P2 è semidecidibile?

Esercizio 2 (8 punti)

Definire in logica del prim'ordine un predicato binario *prefisso* che è vero se e solo se entrambi i suoi argomenti sono stringhe costruite sull'alfabeto A={*a*,*b*} e il primo argomento è un prefisso <u>proprio</u> del secondo. Ad esempio:

- *prefisso*(*a*,*b*) è falso;
- *prefisso*(*ab*,*aba*) è vero;
- *prefisso*(*ab*,*ab*) è falso (*ab* non è un prefisso proprio di *ab*);
- *prefisso*(*a*,*ac*) è falso (le stringhe non sono entrambe costruite sull'alfabeto A).

Oltre ai soliti connettivi, quantificatori, variabili e punteggiatura, si può fare uso <u>esclusivamente</u> dei seguenti simboli:

- le costanti *a* e *b*, che indicano i caratteri alfabetici;
- la costante ε, che indica la stringa vuota;
- il predicato = di uguaglianza tra stringhe;
- la funzione unaria *testa*, che restituisce il primo carattere del suo argomento ed è definita solo per stringhe non vuote
    - ad esempio, *testa*(*abb*) = *a*, mentre *testa*(ε) = ⊥
- la funzione unaria *coda*, che restituisce la stringa composta da tutti i caratteri del suo argomento tranne il primo ed è definita solo per stringhe non vuote
    - ad esempio, *coda* (*abb*) = *bb*, mentre *coda* (ε) = ⊥

Soluzioni che usino altri simboli e convenzioni non saranno prese in considerazione. Si possono naturalmente definire altri predicati di comodo, purché rispettino le convenzioni suindicate.

Soluzione

Esercizio 1 (8 punti)

Chiaramente entrambi i problemi sono indecidibili per il teorema di Rice (la proprietà di riferimento è non banale in entrambi i casi).

P2 è semidecidibile (con una opportuna diagonalizzazione delle stringhe da provare in ingresso, se ci sono 101 stringhe diverse accettate, prima o poi lo scopro). Si può, ad esempio, considerare una macchina di Turing che procede come segue. Per un dato indice i, simula il comportamento della i-esima macchina di Turing M_i; partendo da n=0 e m=1, verifica, per ogni stringa x di lunghezza n, se in m passi M_i accetta x, e procede enumerando (n,m) con la solita tecnica diagonale (0,1), (1,1), (0,2), (2,1), (1,2), (0,3), … fino a che non trova 101 stringhe accettate da M_i, se esistono. Si noti che, fissato n, è sempre possibile enumerare tutte le stringhe di lunghezza n in tempo finito. Pertanto, se esistono 101 stringhe di lunghezza al più n' accettate da M_i in un numero di passi al più m', queste saranno prima o poi enumerate con il procedimento sopra illustrato; se invece non esistono, tale procedimento non termina

P1 è il complemento di P2, che è semidecidibile ma non decidibile, pertanto P1 non può essere nemmeno semidecidibile.

Esercizio 2 (8 punti)

Definisco un predicato di comodo che caratterizza le stringhe sull'alfabeto A:

$\forall x\ (stringa(x) \leftrightarrow x=\varepsilon \vee ((testa(x)=a \vee testa(x)=b) \wedge stringa(coda(x))))$

La formula può essere espressa come segue:

$\forall x \forall y\ (prefisso(x,y) \leftrightarrow stringa(x) \wedge stringa(y) \wedge ((x=\varepsilon \wedge y\neq\varepsilon) \vee (testa(x)= testa(y) \wedge prefisso(coda(x),coda(y)))))$