VARIANTI DI TM

Un esempio semplice: le TM che può spostare la testina a dx, sx o non spostarla:

$$\delta' : Q \times \Gamma \to Q \times \Gamma \times \{L, R, S\}$$

Per ogni TM M' con δ' come sopra, $\exists$ TM M che è equivalente col soddisfa la nostra definizione di TM.
Devo gestire le transizioni di M' che non muovono la testina (S). Posso simularle con una sequenza di 2 movimenti L, R.

Made with Goodnotes

Ovvero: $\delta'(q, a) = (p, b, S)$ corrisponde

e $\quad \delta(q, a) = (r, b, L)$ (r stato ausiliario)

$\quad \delta(r, *) = (p, *, R)$

Le TM multi-nastro: ci sono $k \geq 1$ nastri; input sul 1° nastro e ad ogni passo si aggiornano i nastri e le posizioni delle testine sulla base dei simboli in lettura sui diversi nastri:

$$\delta' : Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L, R\}^k$$

TEO Per ogni TM multinastro, ne esiste una singolo nastro equivalente.

Dim. Data M' la TM multinastro, definisco M TM singolo nastro. M tiene traccia delle configurazioni di M' usando un solo nastro. Idea:

(i) Sia $\#$ il simbolo usato per separare i nastri ($\# \notin \Gamma'$); questo ci permette di rappresentare i k nastri su un solo nastro. Se M' usa Γ', M usa $\Gamma = \Gamma' \cup \{\#\}$.

(ii) Per rappresentare le k testine, marchiamo ciascun simbolo in Γ: $\forall\, a \in \Gamma'$ avremo $\dot{a} \in \Gamma$.

Algoritmo di M:

- Configurazione iniziale:

$$\# \underset{\bullet}{w_1} \dots w_n \# \underset{\bullet}{\sqcup} \# \underset{\bullet}{\sqcup} \dots \# \underset{\bullet}{\sqcup} \#$$

- Passo di computazione: Scorriamo il nastro a partire dal primo # e leggo i simboli marcati su ciascun nastro.

Torno indietro ed eseguo una seconda scansione del nastro aggiornando il contenuto dei k nastri "virtuali" coerentemente con δ'.
Nel fare questo, se la testina su uno dei k

nastro deve spostarsi su #, allora M preventivamente sposta a Dx il contenuto del nastro di 1 posizione.

- Appena M' accetta / rifiuta, M fa lo stesso. ∎

COR. L è TURING-RIC/DEC sse ∃ TM multinastro che la RIC./DEC.

Le TM NON-DETERMINISTICA: Arricchiamo il modello con il NON-DETERMINISMO.

$$\delta : Q \times \Gamma \to \mathcal{P}(Q \times \Gamma \times \{L, R\})$$

Ovvero ci sono diversi rami di computazione parallelo. La NTM accetta se esiste almeno un ramo che accetta. Uno degli altri rami potrebbe anche avere lunghezza infinita.

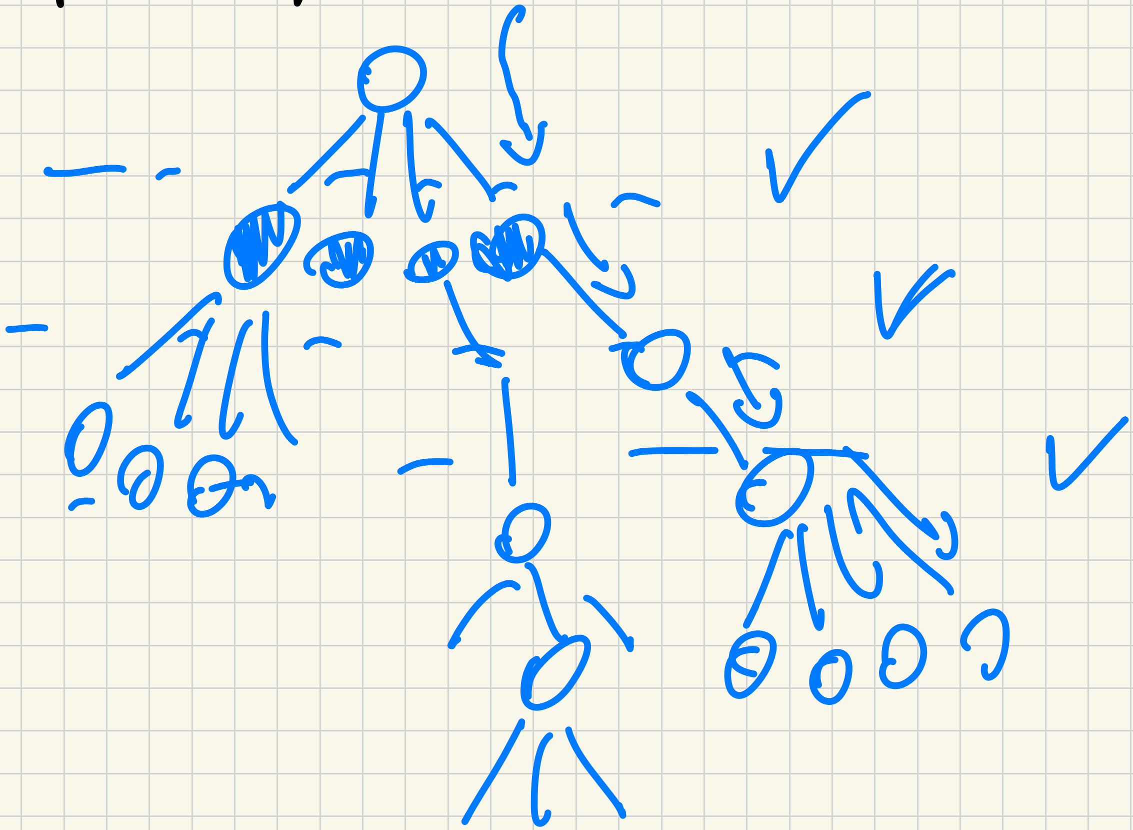
TEO Per ogni NTM N, esiste una TM M equivalente.

Dim. La NTM N su input w ha tanti cammini, ognuno corrisponde col un insieme di scelte non-deterministiche. La TM M deve esplorarli tutti, se uno accetta può accettare.

Difficoltà: Se M esplora i cammini in profondità, può entrare in loop.

Made with Goodnotes

Viceversa, il esplora i cammini in altezza ovvero prima tutti i cammini fino ad altezza 1, poi quelli di altezza 2, etc.



Idee algoritmiche: M userà 3 nastri.

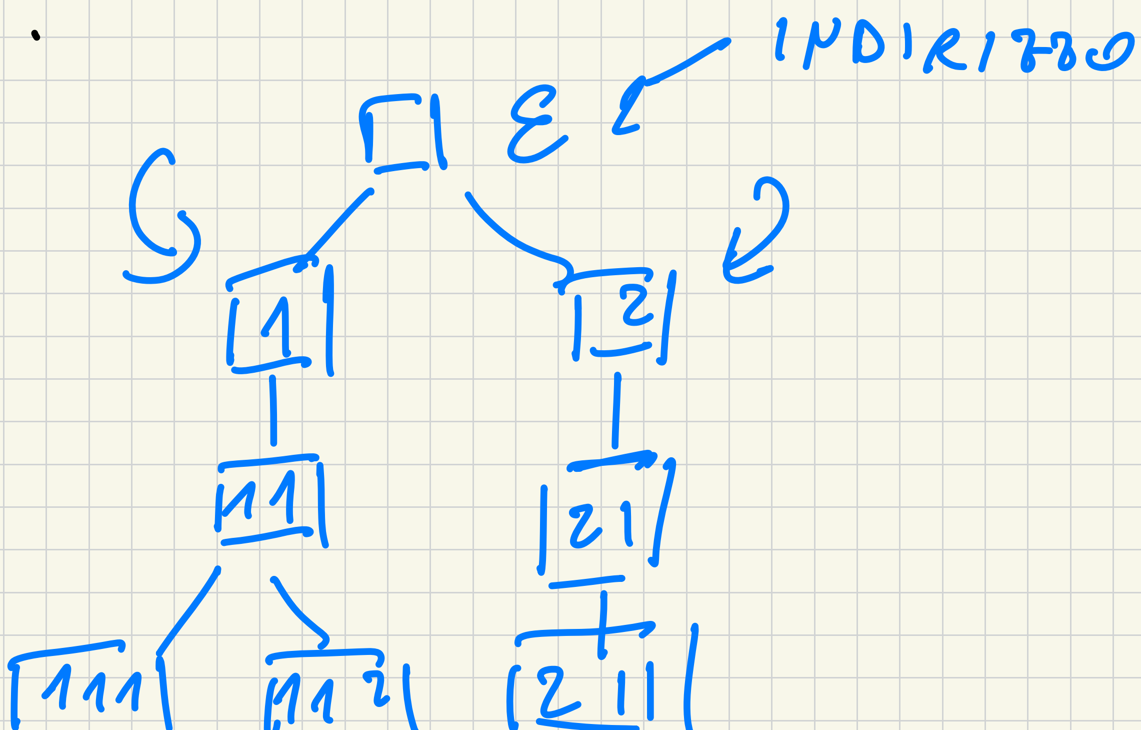
1) Nastro con input, e non cambia mai

2) Nastro di lavoro: mi serve per esplorare un cammino.

3) L'indirizzo del nodo corrente dell'albero che sto esplorando.

$b = 2$

↪ max # di scelte non det.



STEP(M, w, i): usando l'input su nastro 1, simula il cammino di computazione di N che parte dalla radice al nodo con indirizzo i, usando il nastro 2 come nastro di lavoro.
Se N accetta, accetto.
Altrimenti incremento l'indirizzo e ripeto.

Enumeratore: una TM con stampante collegata che produce in output stringhe.
Lo posso vedere come generatore di linguaggi.
Le stringhe possono essere generate in qualsiasi

ordine e con ripetizioni.

TEO Un linguaggio è Turing-riconoscibile sse esiste enumeratore che lo genera.

DIM. ($\Rightarrow$) TM simula enumeratore E.
Assumiamo di avere enumeratore E. La TM M con input w esegue E: ogni volta che E restituisce un output lo confronto con w e accetto quando è proprio w.

($\Leftarrow$) Enumeratore simula TM. Assumo una TM M con linguaggio L(M) su alfabeto Σ. Tutte le stringhe sono Σ^*, di queste E deve enumerare solo quelle in L(M).

Idea: $\Sigma^* = \{ \varepsilon, s_1, s_2, s_3, \dots \}$

$\mathcal{E}$: Ignora l'input per M rispetto

FOR $i = 1, 2, 3, \dots$ (loop infinito)

    FOR $j = 1, \dots, i$

        Simula M con input s_j per i passi

        Se M accetta s_j, stampa s_j $\blacksquare$

(IN) DECIDIBILITÀ.

Abbiamo visto come esistono linguaggi NON regolari, e context-free. Ci chiediamo se tutti i linguaggi sono TURING =

DECIDIBILI / RICONOSCIBILI.

Vedremo che non è così!

Problema di Hilbert: Se esiste un algoritmo che determina se un polinomio del tipo

$$6x^3yz^2 + 3xy^2 - x^3 - 10$$

ha radice intera.

Codifica: Dato un oggetto O (es. polinomio, DFA, CFG, TM), denotiamo con $\langle O \rangle$ la sua codifica binaria.