Lezione 9

indecidibilità

Dalla Lezione 8 sappiamo che:

-le TM riconoscono i linguaggi RE
-le TM che si fermano sempre riconoscono i linguaggi ricorsivi

---dimostreremo che RE $\supset$ ricorsivi

---e che esistono linguaggi che non sono RE

visto il rapporto tra TM e computer reali tutto questo vale anche per i computer reali

Codifichiamo le TM in modo sistematico

--numeriamo le strighe di 0/1, per stringa w, 1w è il suo numero (indice)

1=ε, 10=0, 11=1, 100=00,101=01,110=10,...

w6=110=2

e w37 a quale stringa corrisponde ?

codifica delle TM: ogni TM con input {0,1} viene rappresentata da una stringa binaria w_i, allora sarà la TM Mi

Rappresentazione:

--gli stati sono q1,q2,...qr, per r>0. q1 iniziale e q2 finale (unico e senza transizioni). Li rappresentiamo come interi

--simboli di nastro X1...Xs, dove X1=0, X2=1 e X3=B, gli altri, X4...Xs hanno indici qualsiasi.

--L/R D1/D2

visto che alcune cose sono arbitrarie abbiamo molte codifiche possibili per una TM

la regola di transizione
$\delta(qi,Xj)=(qk,Xn,Dm)$

è rappresentata da $0^i10^j10^k10^n10^m$

non ci sono mai due o più 1 consecutivi

il codice di una TM sarà: $C_1 11 C_2 11 C_3 11 \ldots C_{n-1} 11 C_n$

<u>esempio</u>: $M=(\{q1,q2,q3\},\{0,1\},\{0,1,B\}, \delta, q1,B,\{q2\})$ con le seguenti transizioni:
$\delta(q1,1)=(q3,0,R)$, $\delta(q3,0)=(q1,1,R)$, $\delta(q3,1)=(q2,0,R)$, $\delta(q3,B)=(q3,1,L)$

la prima transizione corrisponde a
0100100010100 dato che $\delta(q1,1)=(q3,0,R)$ potrebbe essere:
$\delta(q1,X2)=(q3,X1,D2)$

codifica simile per le altre transizioni

Molti codici per la stessa TM, anche non c'è un ordine delle codifiche delle transizioni

TM → codifica → stringa di 0/1 che sono numerate wi →Mi

ovviamente molte stringhe binarie wi non corrispondono ad alcuna TM. In questo caso Mi è la TM senza mosse che accetta il linguaggio vuoto, L(Mi)=∅

Possiamo costruire una tabella:

	1	2	3	4	…….	
M1	0	1	1	0	……..	
M2	1	1	0	0	……….	
M3	0	0	1	1	………	
M4	0	1	0	1		

……………………………… la riga i dice se Mi accetta wi = Mi

	1	2	3	4	…….	
M1	**0**	1	1	0	……..	
M2	1	**1**	0	0	……….	
M3	0	0	**1**	1	………	
M4	0	1	0	**1**		

L_d è l'insieme delle w_i t.c. w_i non è in $L(M_i)$

è il complemento della diagonale della tabella

Teorema nessuna TM accetta L_d

Dimostrazione: Suppponiamo che esista M che accetti L_d. L_d è un linguaggio su 0/1, quindi M sarà M_i per qualche riga i della tabella. Ma allora M_i accetta w_i sse M_i non accetta w_i.
Contraddizione!
L_d non è RE.

es. 9.1.3 (a)* (b)

Linguaggi ricorsivi sono quei linguaggi L per cui esiste una TM M tale che L=L(M) e precisamente:

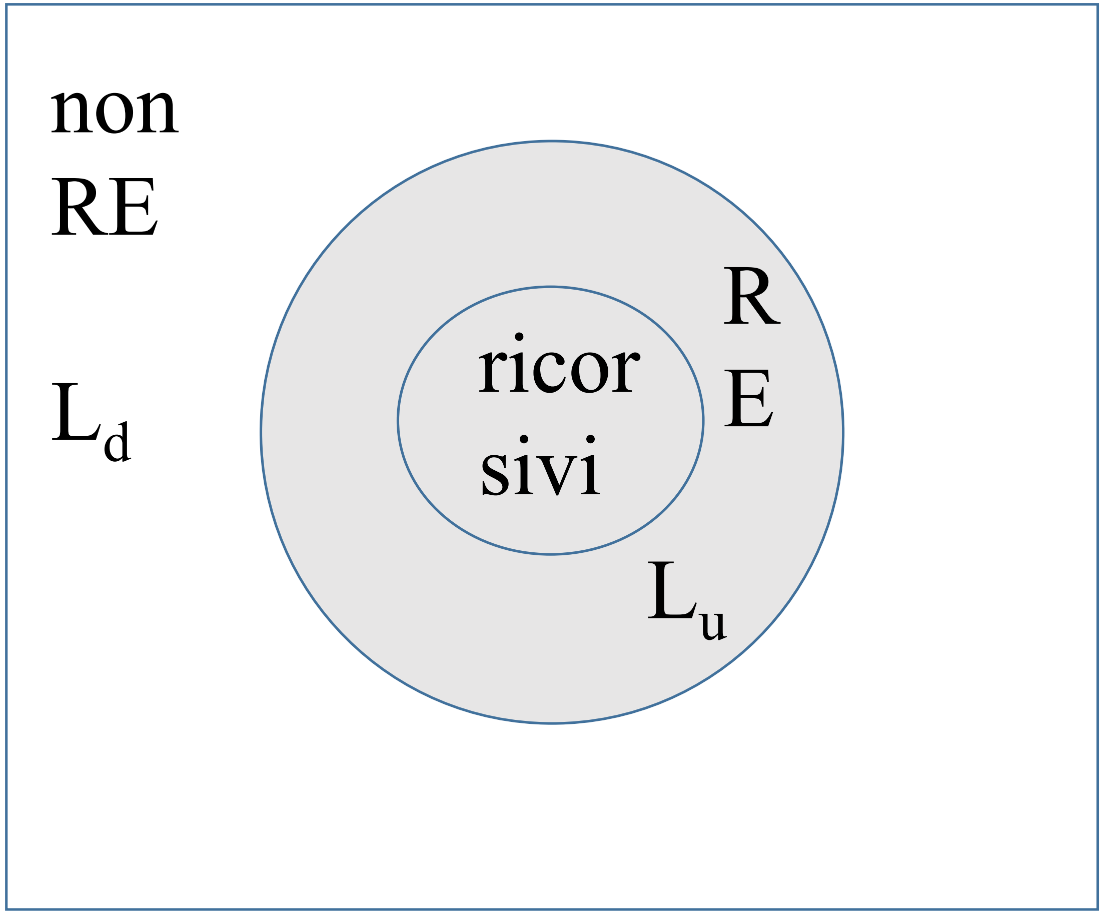
--se w è in L allora M accetta e si ferma

--se w non è in L, allora M si ferma in uno stato non accettante.

Una tale TM corrisponde alla nozione di <u>algoritmo</u>

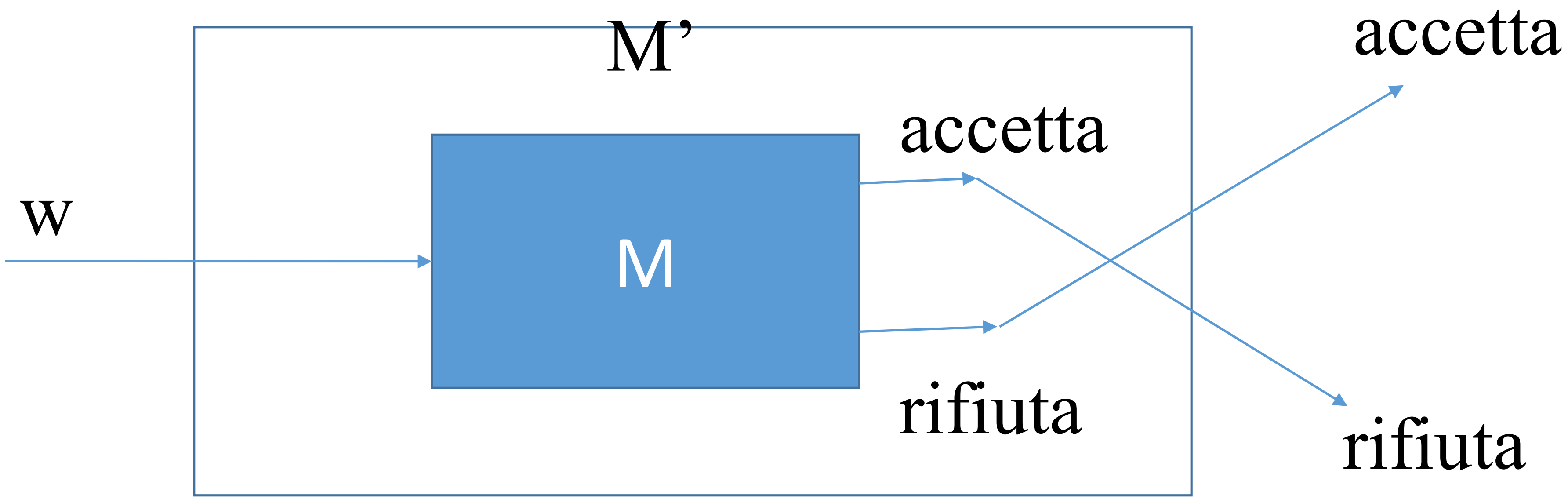
Se «vediamo» L come un problema, allora diremo che l’algoritmo M decide L e che L è decidibile

se non esiste algoritmo per L, allora L è indecidibile la situazione è questa:



Teorema: se L è ricorsivo, allora anche il suo complemento comp(L) lo è

dimostrazione: sia L=L(M) per una TM M che si ferma sempre. E' facile usare M per costruire una TM M' che decide comp(L):



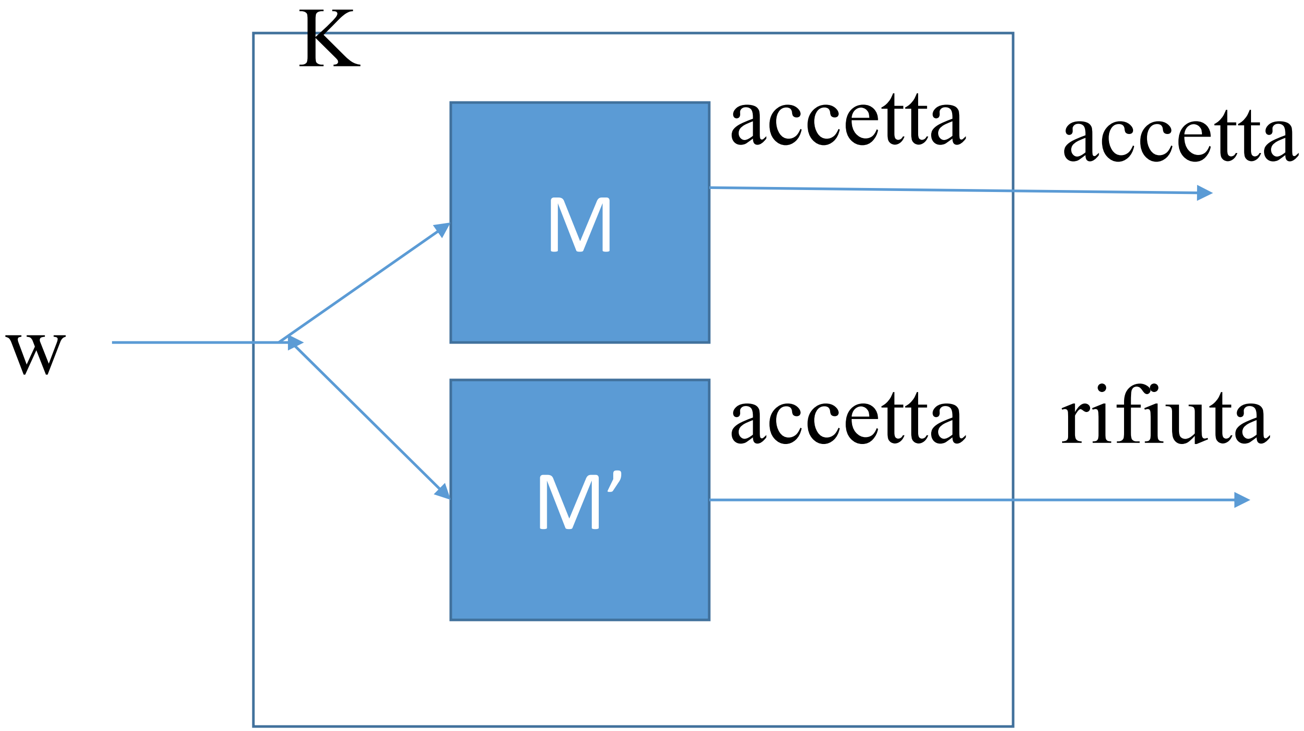
in dettaglio, M’ è ottenuto da M come segue

--gli stati finali di M diventano non finali;
--si aggiunge un nuovo stato finale r, senza transizioni;
--per ogni coppia (s,X) con s stato non finale di M e X simbolo del nastro, tale che $\delta(s,X)$ è indefinita, si aggiunge
$\delta(s,X)=(r,X,R)$

Dato che M si ferma sempre, anche M’ lo fa e M’ accetta le stringhe che M rifiuta, cioè comp(L).

Teorema: se un linguaggio L e comp(L) sono entrambi RE, allora sono entrambi ricorsivi.

Dimostrazione: l'idea è semplice



K simula M e M’ in parallelo su w, n passi una, e poi n passi l’altra.
ricordandosi stato e posizione sul nastro diviso in 2 tracks

per L e comp(L) ci sono solo 4 possibilità:
-- L e comp(L) sono entrambe ricorsive
-- L e comp(L) sono entrambe non RE
-- L è RE ma non ricorsivo e comp(L) non RE
-- comp(L) è RE ma non ricorsivo e L è non RE

$comp(L_d)$ è RE

$comp(L_d) = \{w_i \mid M_i \text{ accetta } w_i\}$

una TM M con w_i come input può simulare M_i su w_i

se M_i accetta allora M accetta, se M_i rifiuta fermandosi, rifiuta, e se continua a calcolare, anche M continua a calcolare

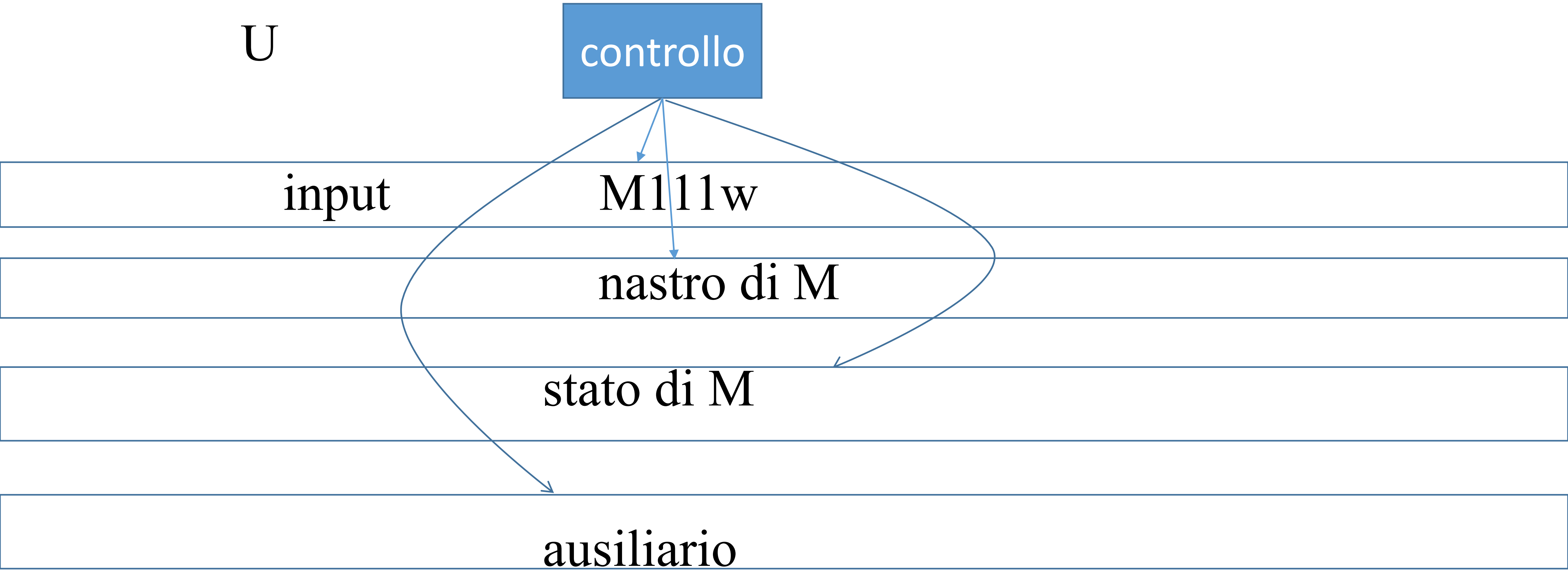
tra poco vedremo meglio come una TM ne può simulare un'altra

Linguaggio universale

L_u={ (M,w) | M è una TM con input 0/1, w è una stringa di 0/1 e w è in L(M)}

mostriamo che esiste una TM U detta TM universale tale che $L(U)=L_u$

visto che U ha input di 0/1 U = M_j per qualche j


U
controllo
input
M111w
nastro di M
stato di M
ausiliario

come opera U:

1) esamina l'input per controllare che la prima parte rappresenti una TM, altrimenti stop con rifiuto
2) prepara input sul secondo nastro 0 = 10, 1 = 100, B=1000
3) scrive $q_1 = 0$ sul terzo nastro e posiziona la testina sul margine sinistro dell'input
4) per simulare una mossa di M, cerca sul primo nastro una transizione che inizia con lo stato corrente 0^i e con simbolo letto 0^j, supponiamo che la transizione sia $0^i10^j10^k10^x10^m$

allora cambia lo stato sul terzo nastro in 0^k, cambia 0^j sul nastro 2 con 0^x e sposta la testina del nastro 2 L/R a seconda di m

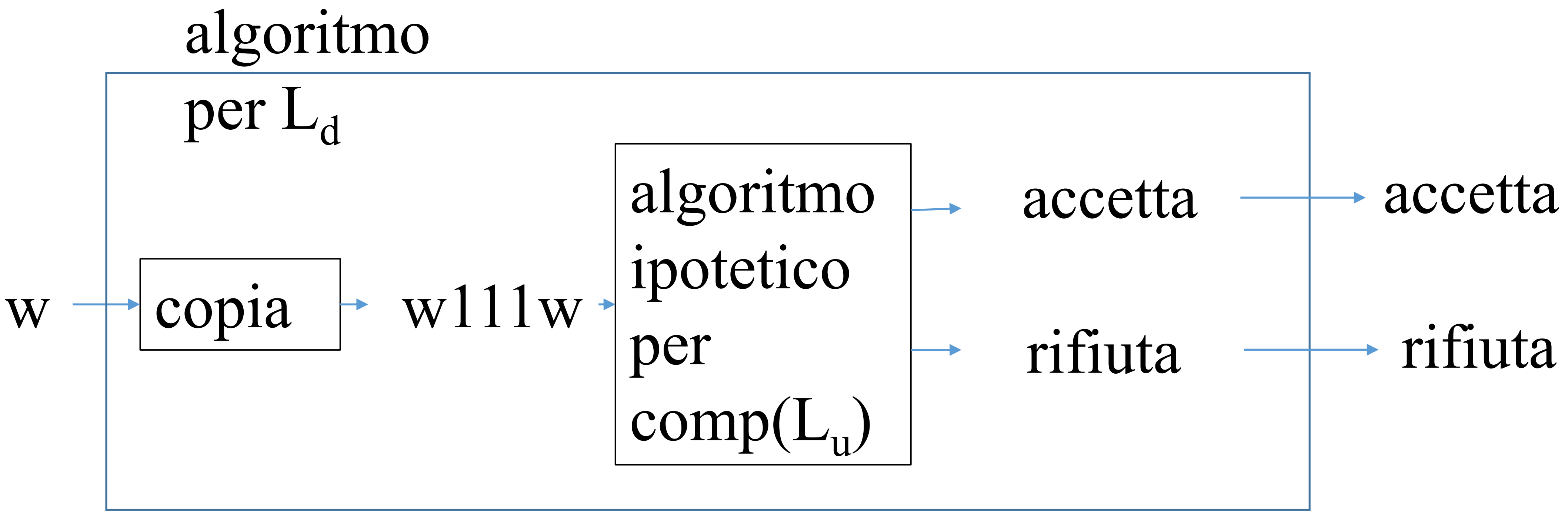
5) se non ci sono transizioni applicabili (M si blocca), allora U si blocca

6) se M entra nello stato finale q_2, U accetta

<u>U simula M su w e accetta (M,w) sse M accetta w, quindi L_u è RE</u>

Teorema: L_u non è ricorsivo

Dimostrazione: se L_u fosse ricorsivo anche comp(L_u) sarebbe ricorsivo e con una TM che riconosce comp(L_u) potremmo decidere L_d che non è RE. Contraddizione!



dal fatto che un problema P è indecidibile, possiamo derivare che un altro problema P’ è indecidibile

riduciamo P -> P’

ogni istanza I di P viene trasformata in un’istanza h(I) di P’ tale che la I ha risposta si sse h(I) ha risposta si

Teorema: se esiste una riduzione P -> P’ allora
a)se P è indecidibile anche P’ lo è
b)se P non è RE anche P’ non è RE

dimostrazione:
a) se fosse possibile decidere P’ allora potrei decidere anche P

per ogni istanza I di P costruiamo h(I) di P’, la decidiamo e usiamo la stessa risposta per I

quindi P non può essere indecidibile

b) se P’ è RE allora esiste una TM M che da le risposte giuste per le istanze SI di P’, ma allora ci sarebbe anche per P:
per ogni istanza I → h(I), con M decido le risposte SI e le uso per I
quindi decido le risposte SI di P, quindi anche P sarebbe RE

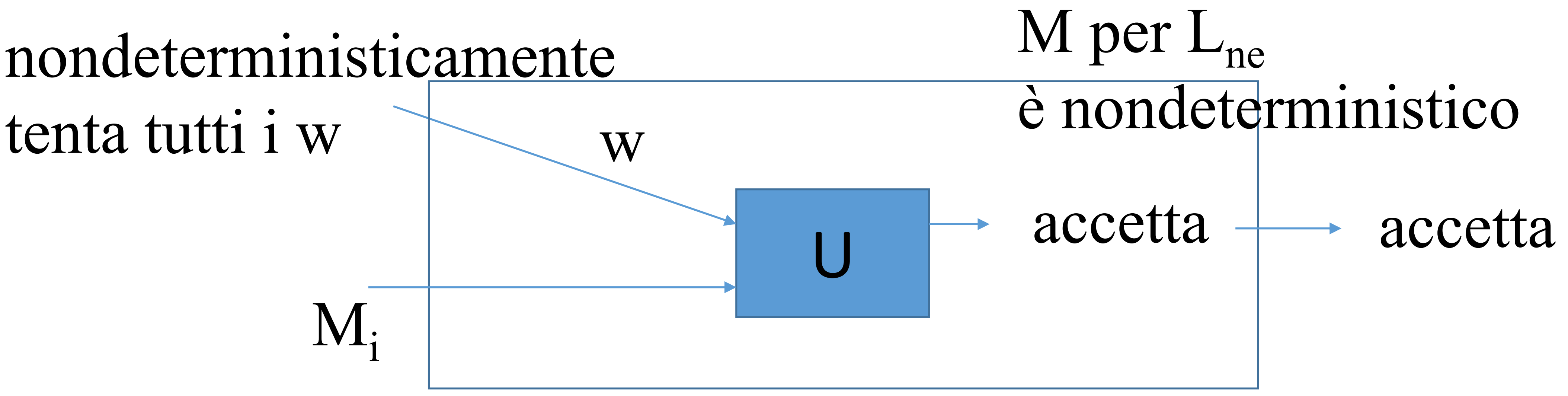
esempi di riduzioni:

$L_e = \{M \mid L(M) = \emptyset\}$

$L_{ne} = \{M \mid L(M) \neq \emptyset\}$

visto che le TM sono rappresentate da sequenze binarie, si tratta di linguaggi sulle stringhe binarie. Sono uno il complemento dell'altro

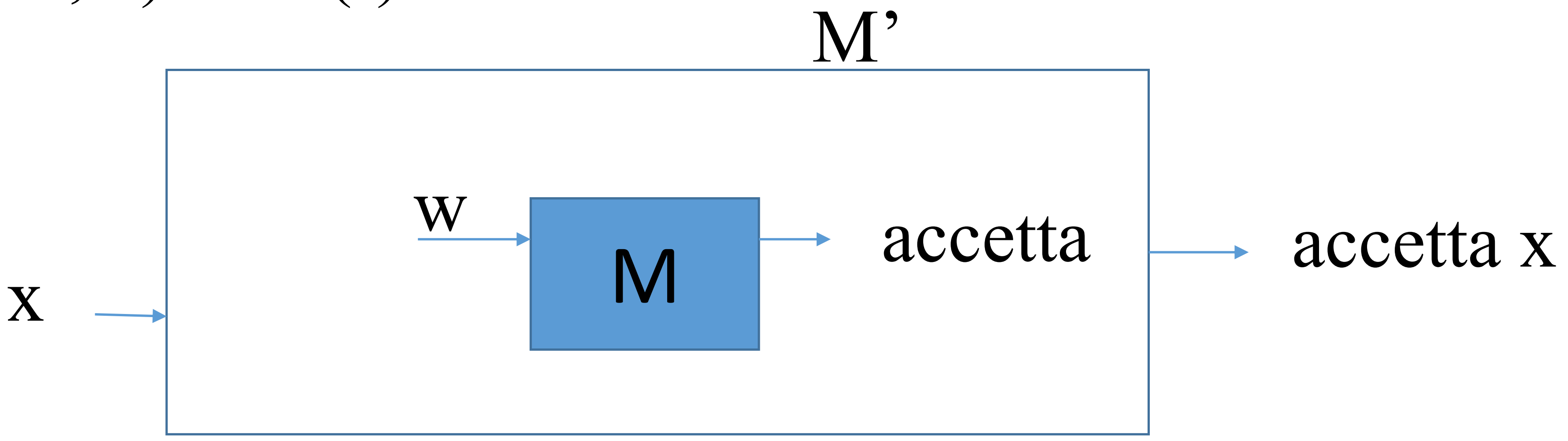
Teorema 9.8: L_{ne} è RE



Teorema 9.9. L_{ne} non è ricorsivo

<u>Dimostrazione.</u> Usiamo la riduzione L_u -> L_{ne}

I=(M,w) -> h(I) = M'



M accetta w sse M' accetta x e quindi

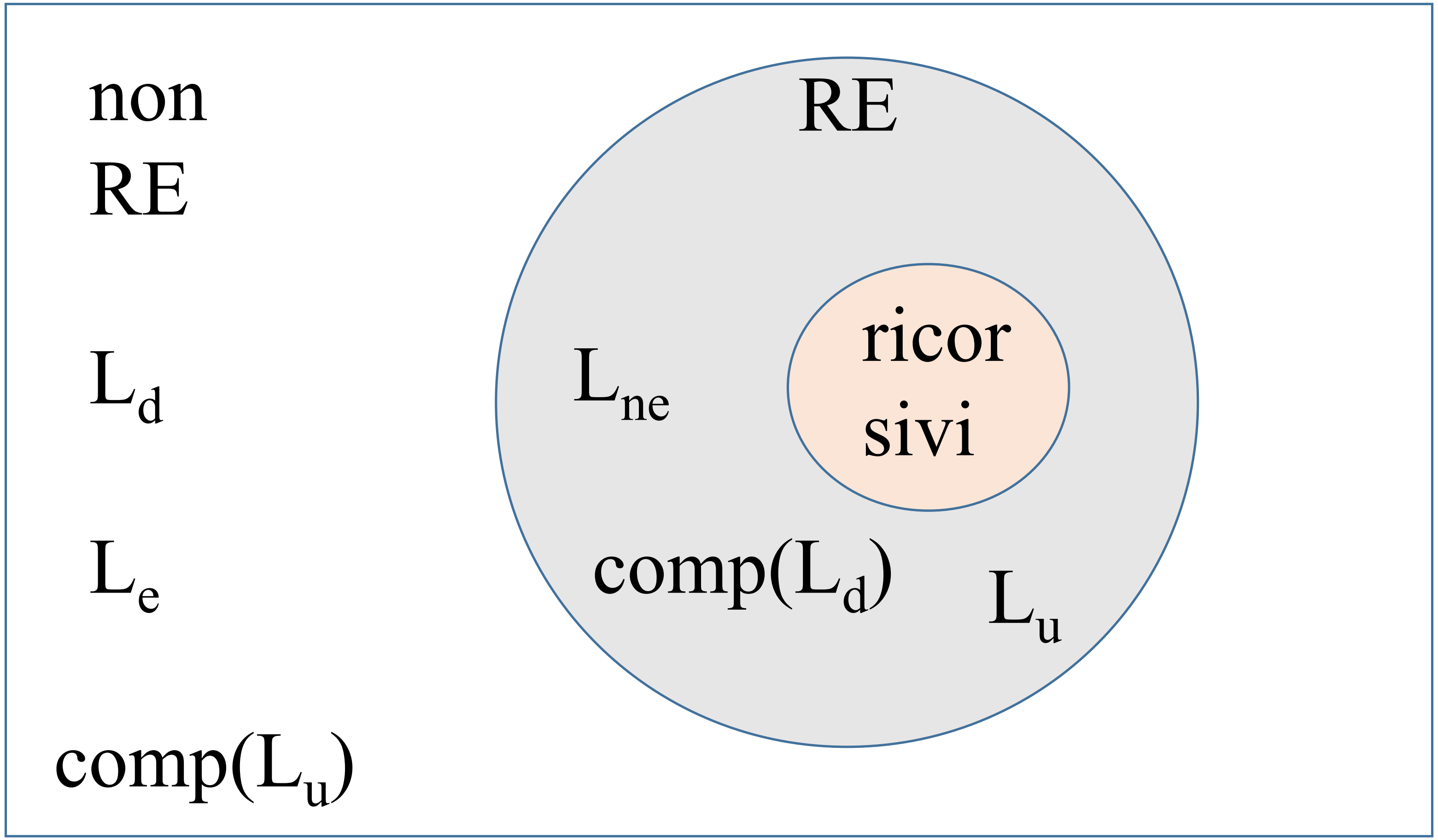
(M,w) è in L_u sse M' è in L_{ne}

L_{ne} è RE e non ricorsivo

Teorema 9.10. L_e non è RE

Dimostrazione. Se L_e fosse RE allora sia L_e che L_{ne} sarebbero ricorsivi => assurdo.

la situazione ora è questa:



Linguaggi RE = TM

Proprietà P dei linguaggi RE = insieme dei linguaggi RE che la soddisfano. Come facciamo a riconoscere insiemi di linguaggi? Per ogni linguaggio RE c'è (almeno) una TM che lo riconosce

P=> L_P={M | M riconosce L che soddisfa P}
L_P è insieme di stringhe binarie

P è <u>banale</u> se L_P è $\emptyset$ oppure contiene tutti i linguaggi RE

Esempi di proprietà non banali dei linguaggi RE

contenere tutti i palindromi

essere CF

essere regolari

essere infiniti

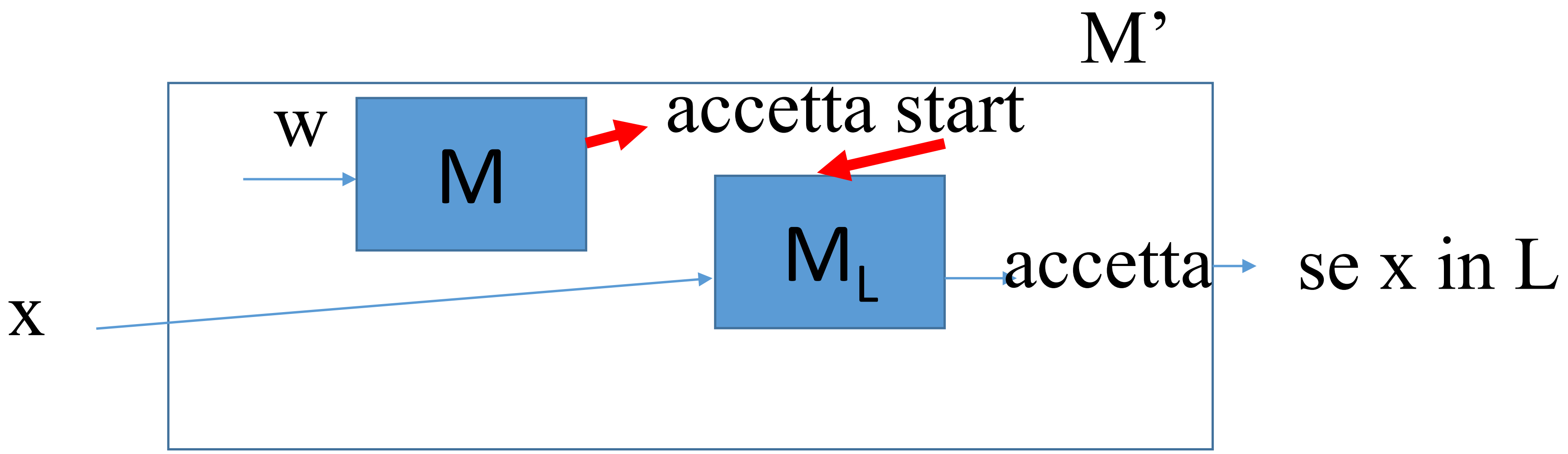
essere finiti

Teorema di Rice

Tutte le proprietà non banali dei linguaggi RE sono indecidibili.

<u>Dimostrazione</u>. Sia P una proprietà non banale. Supponiamo che $\emptyset$ non sia in L_P. Sia L in L_P supponiamo di avere M_L che accetta L.

$L_u \to L_P$ $\quad$ $I=(M,w) \to h(I) = M'$



(M,w) è in L_u sse M' è in L_P

se L_P fosse decidibile => L_u sarebbe decidibile

E se $\emptyset$ è in L_P?
Consideriamo comp(P). E' l'insieme dei linguaggi RE che non soddisfano P, quindi, per quanto dimostrato prima, comp(P) è indecidibile. Cioè $L_{comp(P)}$ è indecidibile, ma $L_{comp(P)}$ = comp(L_P). Per cui, se L_P fosse decidibile allora lo sarebbe anche $L_{comp(P).}$ Contraddizione.

conseguenze del Teorema di Rice:

--Linguaggio vuoto?
--Linguaggio finito?
--Linguaggio regolare?
--Linguaggio libero dal contesto?

sono tutti problemi indecidibili