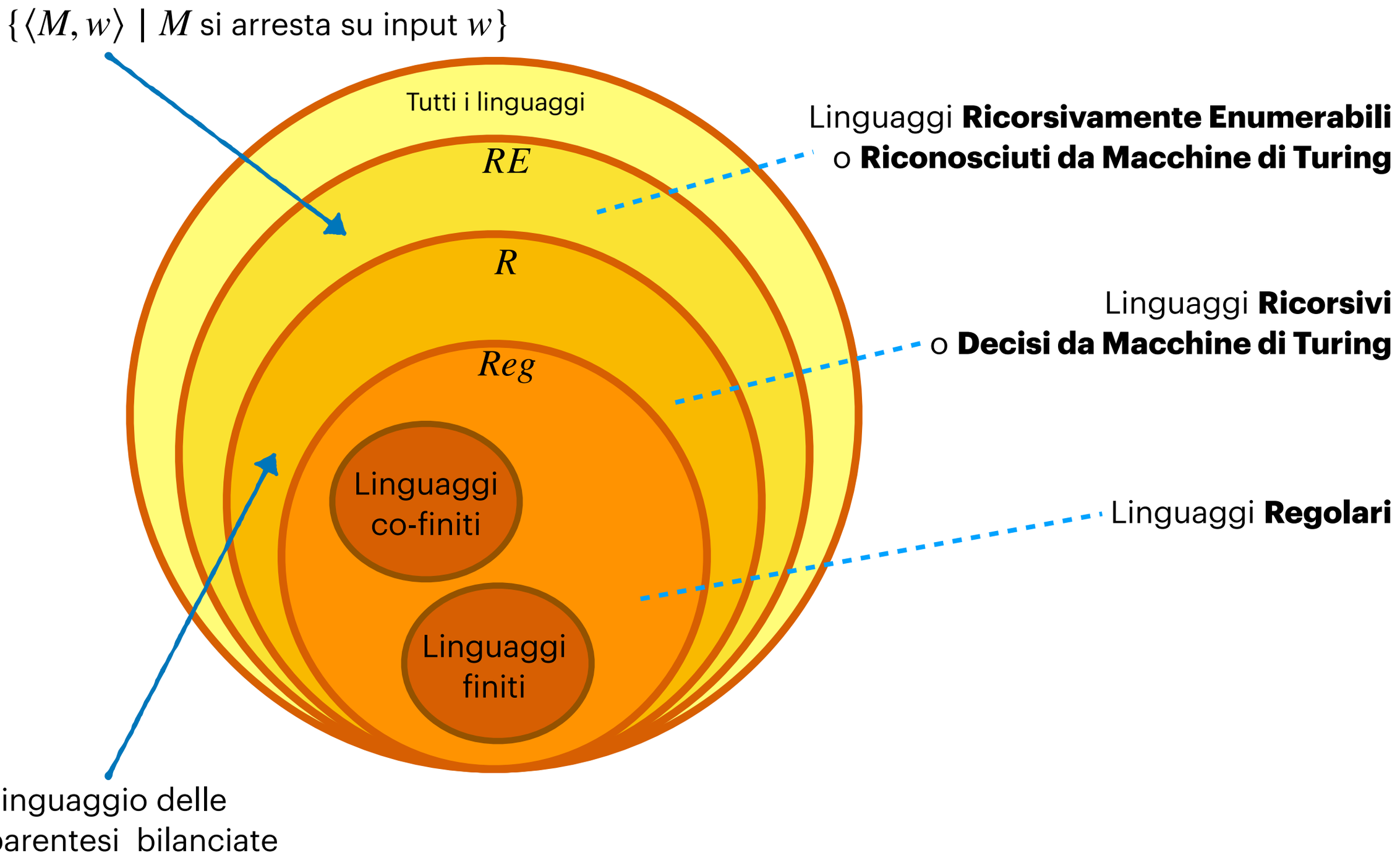
Computabilità, Complessità e Logica

Lezione 5

Classi di linguaggi

Relazione tra RE, R e Reg



Macchina di Turing universale

Possiamo costruire una macchina di Turing che simula altre macchine di Turing?

L'idea di avere una macchina che ne simula un'altra data la sua descrizione appare nell'articolo di Turing del 1936

Idea di base:

- Tenere traccia dello configurazione della macchina simulata
- Per vedere quale è la configurazione successiva si consulta la tabella di transizione della macchina simulata

Proprietà di chiusura

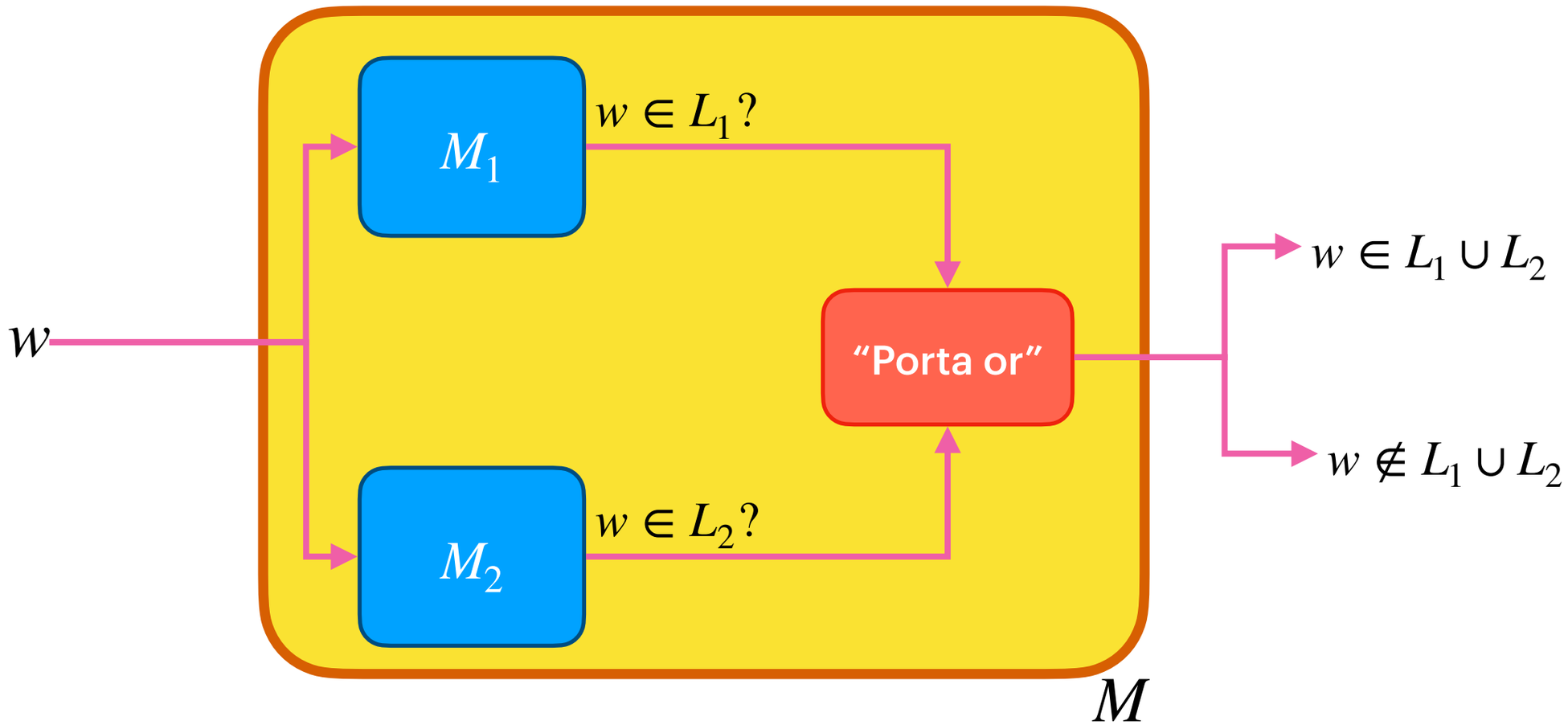
Potremmo chiederci se la classe dei linguaggi decisi da MdT e riconosciuti da MdT sono chiuse rispetto ad alcune operazioni insiemistiche:

- Unione
- Intersezione
- Complementazione

Per semplicità assumiamo la tesi di Church-Turing e descriveremo gli algoritmi in come pseudocodice/ procedura meccanica, sapendo che possono poi essere implementati da una macchina di Turing

Unione di linguaggi decidibili

Dati due linguaggi L_1 e L_2 **decisi** da due macchine di Turing M_1 e M_2 possiamo costruire un macchina di Turing M che **decide** $L_1 \cup L_2$:



Unione di linguaggi decidibili

Perché possiamo costruire M?

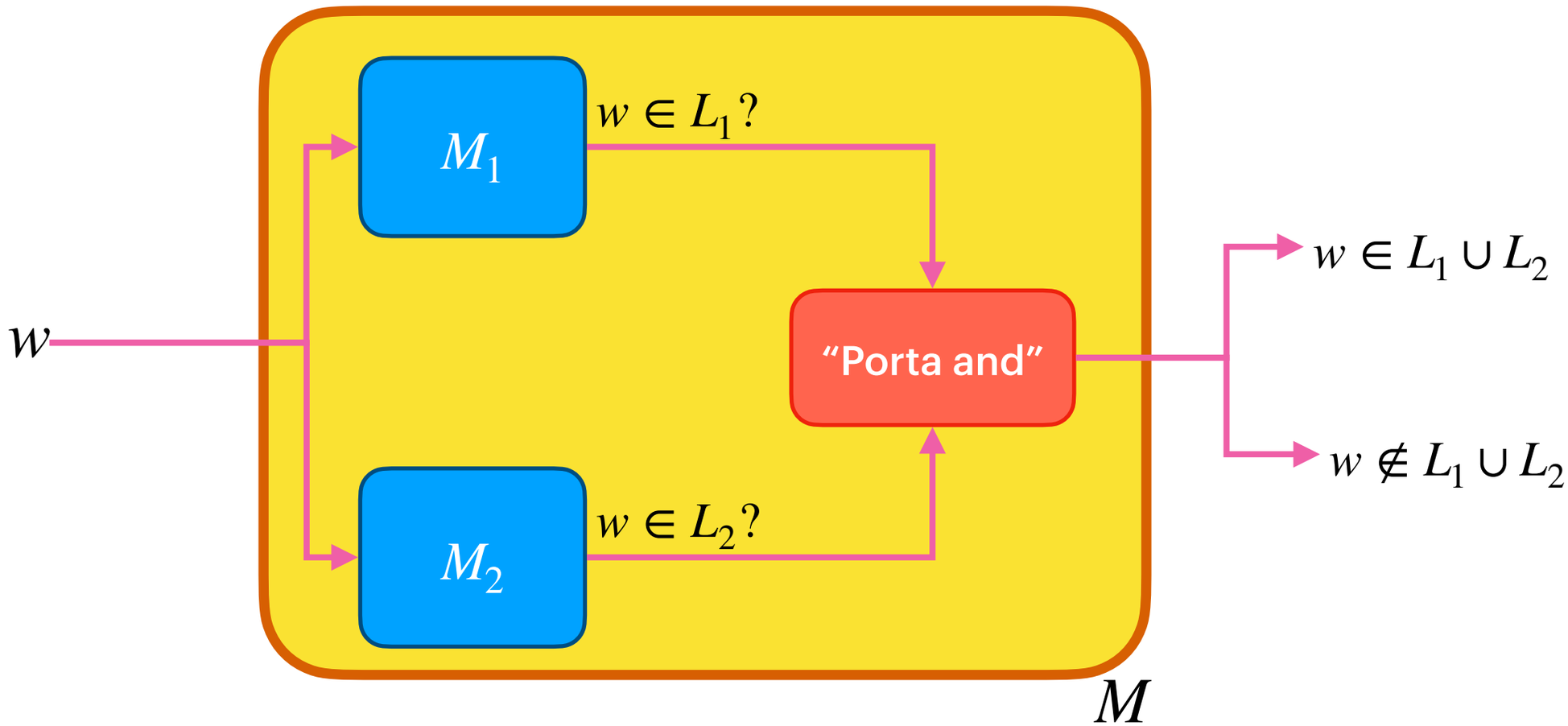
Possiamo simulare prima M_1 su input w e poi M_2 su input w e combinare i due risultati con una disgiunzione (porta OR)

Siamo sicuri che M **decida** $L_1 \cup L_2$?

Se M si arresta allora il risultato è corretto. Per essere sicuri che M si arresti ci basta notare che dato che M_1 e M_2 **decidono** L_1 e L_2 allora queste (e la loro simulazione) si arrestano sempre e quindi anche M si arresta sempre.

Intersezione di linguaggi decidibili

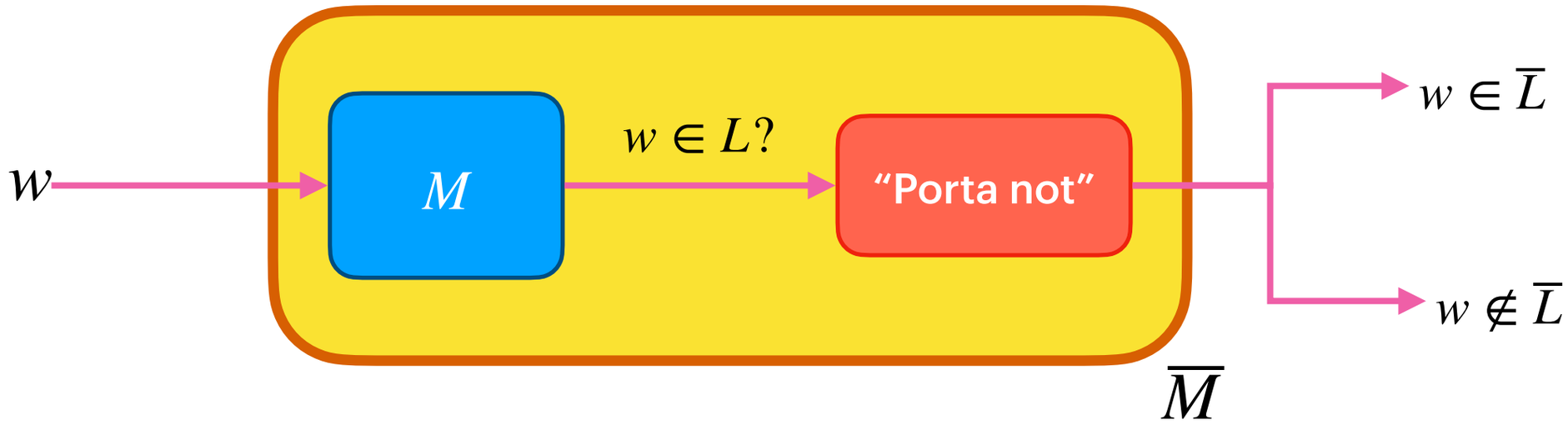
Dati due linguaggi L_1 e L_2 **decisi** da due macchine di Turing M_1 e M_2 possiamo costruire un macchina di Turing M che **decide** $L_1 \cap L_2$:



Valgono poi gli stessi ragionamenti fatti per l’unione

Complemento di linguaggi decidibili

Dati un linguaggio L **deciso** da una macchine di Turing M
possiamo costruire un macchina di Turing $\overline{M}$ che **decide** $\overline{L} = \Sigma^\star - L$:



Come per unione e intersezione, la macchina risultante
si arresta su ogni input perché sappiamo

che M si arresta su ogni input (ci basta invertire il risultato)

E per i linguaggi riconosciuti da MdT?

Non possiamo usare direttamente le stesse costruzioni usate per i linguaggi decidibili.

Perché?

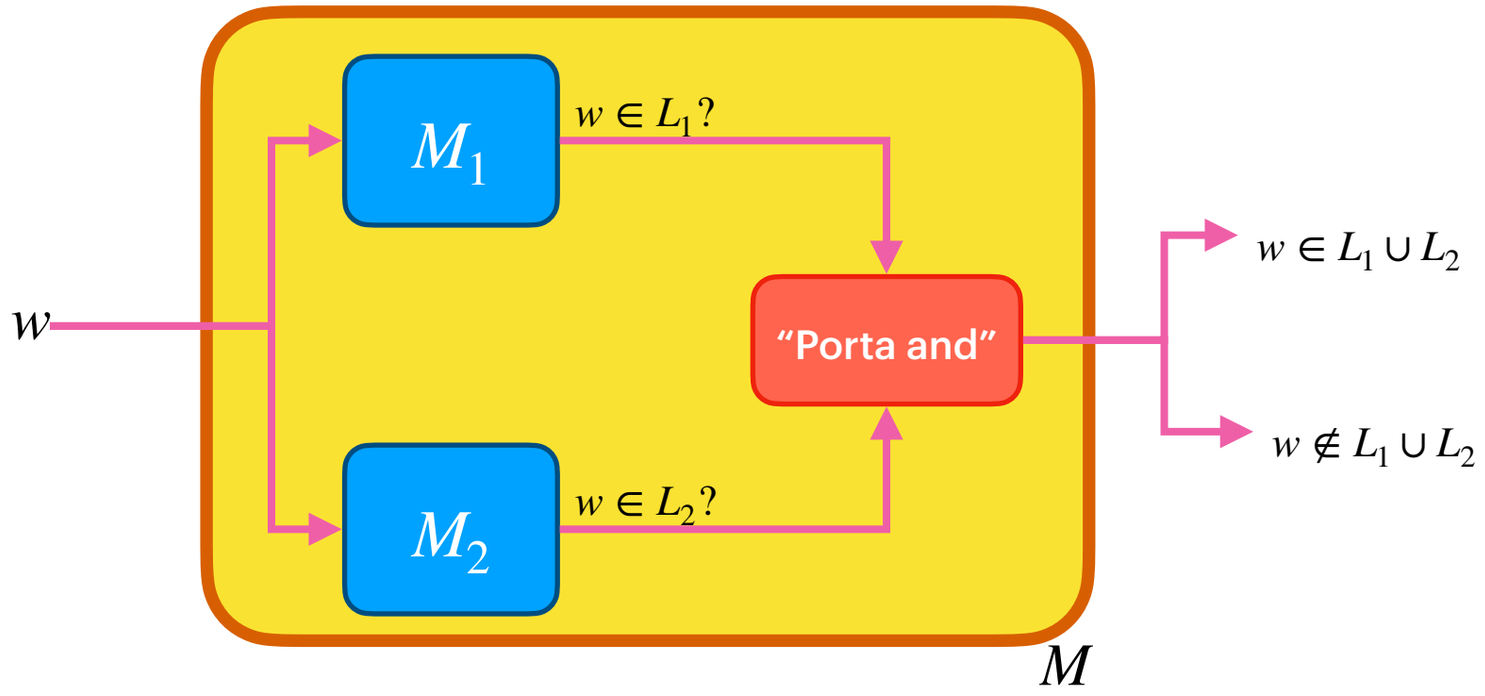
Dato un linguaggio L riconosciuto da una MdT M e una parola w, se $w \notin L$ **non** possiamo essere sicuri che M si arresti

Questo “rompe” due costruzioni:

- Unione (ma è *“riparabile”*)
- Complemento (non *“riparabile”*)

Intersezione di linguaggi riconosciuti da MdT

Dati due linguaggi L_1 e L_2 **riconosciuti** da due macchine di Turing M_1 e M_2 possiamo costruire un macchina di Turing M che **riconosce** $L_1 \cap L_2$:



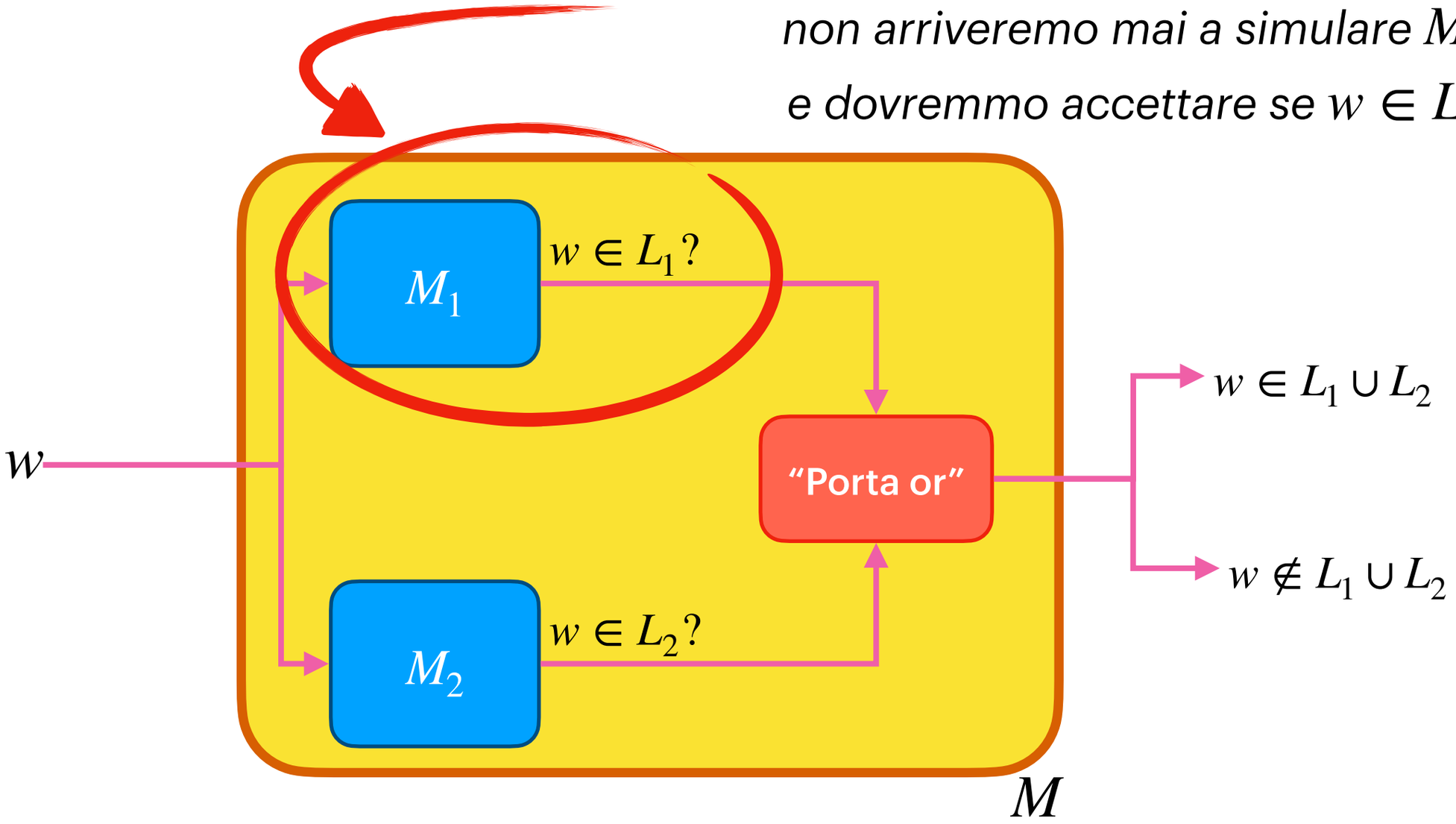
*Se una tra M_1 e M_2 **non** si arresta su input w (e la loro simulazione **non** si arresta) non è un problema perché allora sicuramente $w \notin L_1 \cap L_2$*

Infatti $w \in L_1 \cap L_2$ implica che sia M_1 che M_2 si arrestino (accettando) su input w

Unione di linguaggi riconosciuti da MdT

Dati due linguaggi L_1 e L_2 **riconosciuti** da due macchine di Turing M_1 e M_2 possiamo costruire un macchina di Turing M che **riconosce** $L_1 \cup L_2$?

Se simuliamo M_1 ma $w \notin L_1$ e M_1 non si arresta
non arriveremo mai a simulare M_2
e dovremmo accettare se $w \in L_2$



Simulazione per i passi

Come possiamo "aggiustare" la costruzione per l'unione?

Simuliamo la macchina M_1 per un numero $i \in \mathbb{N}$ di passi, poi simuliamo M_2 per lo stesso numero di passi:

Se una delle due accetta allora accettiamo

Se nessuna delle due accetta allora le simuliamo per $i + 1$ passi

Se $w \in L_1$ o $w \in L_2$ allora dopo un numero finito di passi accetteremo (una tra M_1 e M_2 accetterà). Se nessuna delle due accetta non è un problema (possiamo non arrestarci)

La classe coRE

La classe dei linguaggi complementari a RE è detta $coRE$

Quindi $L \in RE$ implica che $\overline{L} \in coRE$ con $\overline{L} = \Sigma^\star - L$

Potrebbe essere che $RE = coRE$?

Mostriamo che se $L \in RE \cap coRE$ allora $L \in R$, ovvero è decidibile.

Quindi $RE \cap coRE = R$ e, dato che $RE \neq R$, $RE \neq coRE$

Quindi il complementare di un linguaggio ricorsivamente enumerabile è

Caratterizzazione di $RE \cap coRE$

Se $L \in RE$ e $\overline{L} \in RE$ allora $L \in R$

Dimostrazione

Se $L, \overline{L} \in RE$ allora esistono due MdT M e $\overline{M}$ che riconoscono rispettivamente L e $\overline{L}$

Sia $w \in \Sigma^\star$, allora:

Se $w \in L$ la macchina M accetta su input w

Se $w \notin L$ la macchina $\overline{M}$ accetta su input w

Caratterizzazione di $RE \cap coRE$

Costruiamo una macchina M' che **decide** L

Su input $w \in \Sigma^\star$ simuliamo per ripetutamente per un numero crescente di passi le due macchine M e $\overline{M}$.

Sappiamo per certo che una delle due si arresterà accettando.

Se la macchina M accetta su input w allora M' accetta.

Se la macchina $\overline{M}$ accetta su input w allora M' rifiuta.

∎

Caratterizzazione di $RE \cap coRE$

Abbiamo quindi mostrato che $RE \cap coRE \subseteq R$

Sappiamo che $R \subseteq RE$

Dato che R è chiuso rispetto alla complementazione si ha che $R \subseteq coRE$

Quindi $R \subseteq RE \cap coRE$

Si ha quindi che $R = RE \cap coRE$.

Quindi i linguaggi decisi da macchine di Turing sono esattamente quelli per cui riconosciuti da MdT il cui complemento è anch'esso riconosciuto da MdT

Proprietà di chiusura

Linguaggi ricorsivi

Chiusi rispetto all’unione

Chiusi rispetto all’intersezione

Chiusi rispetto alla complementazione

Linguaggi ricorsivamente enumerabili

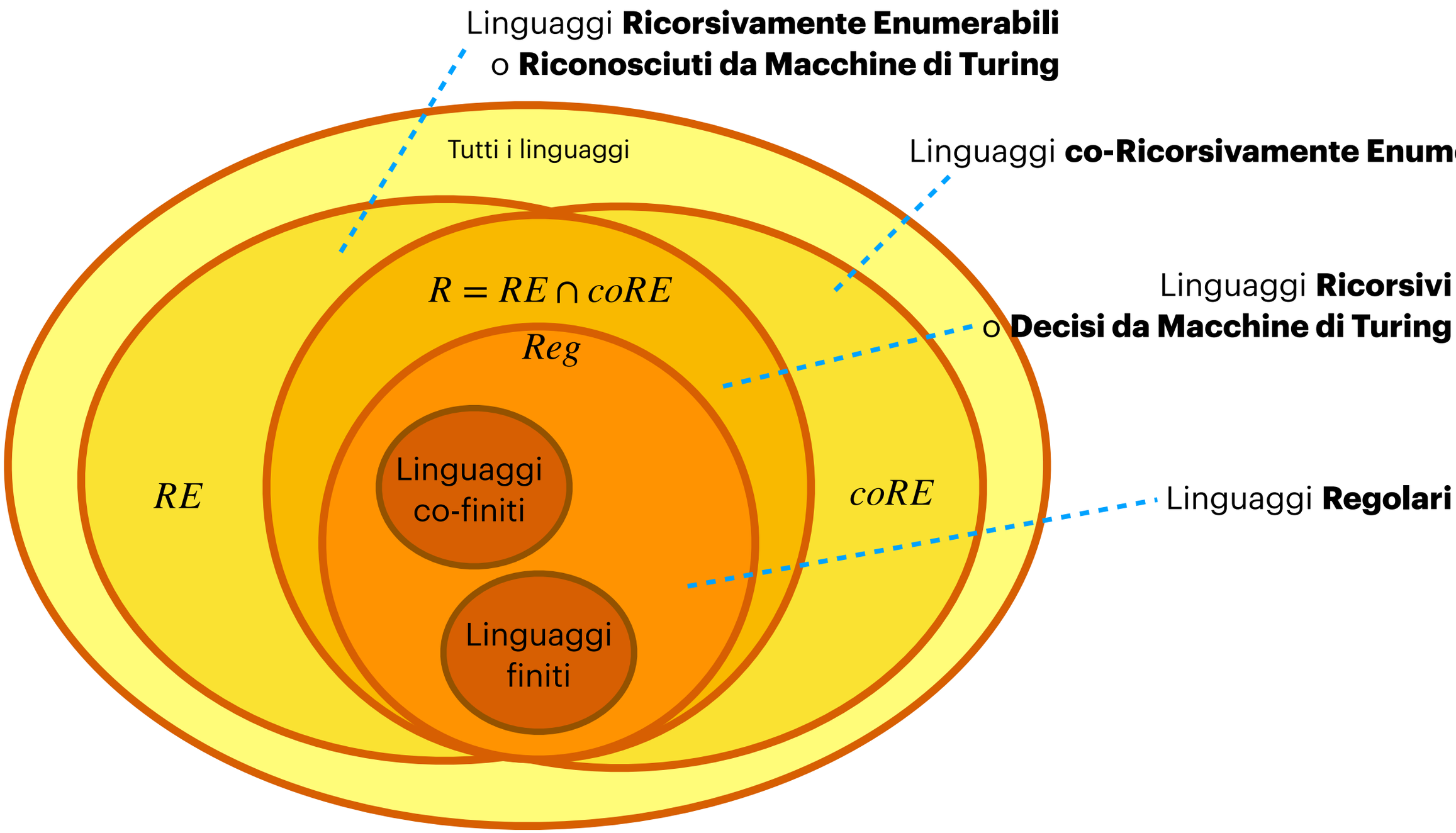
Chiusi rispetto all’unione

Chiusi rispetto all’intersezione

Non sono chiusi rispetto alla complementazione

Classi di linguaggi

Relazione tra RE, coRE, R e Reg



Oltre RE e $coRE$

Esistono linguaggi al di fuori di RE e $coRE$?

Sì, possiamo mostrarlo con un argomento di cardinalità

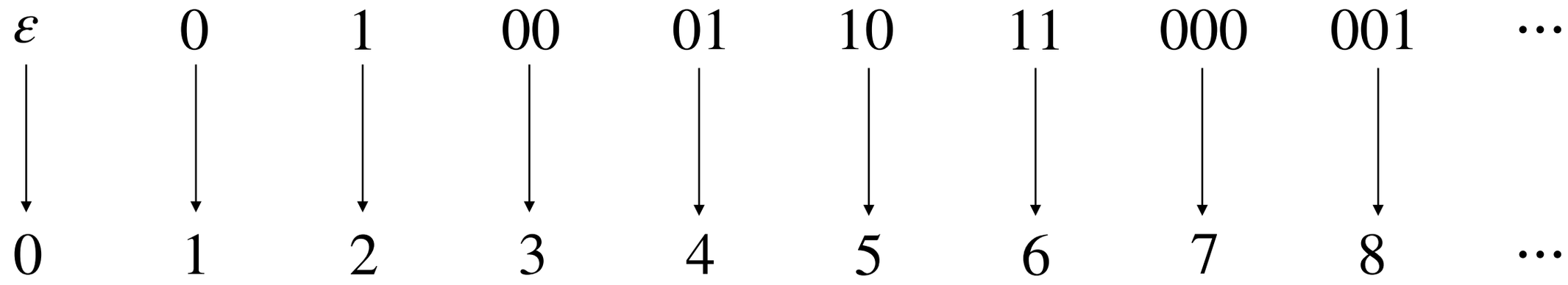
Assunzione: sapere che la cardinalità di $\mathbb{N}$ è inferiore rispetto alla cardinalità di $\mathbb{R}$ (i.e., non esiste una biezione da $\mathbb{N}$ a $\mathbb{R}$ e, dato che $\mathbb{N} \subset \mathbb{R}$ ne segue $\text{card}(\mathbb{N}) < \text{card}(\mathbb{R})$)

Supponiamo di usare l'alfabeto $\Sigma = \{0,1\}$ (ogni altro alfabeto possiamo codificarlo come sequenza di 0 e 1)

Scegliamo una codifica per MdT in binario

Le MdT sono numerabili

Ordiniamo tutti gli elementi di $\Sigma^\star$ per lunghezza e, al loro interno, in ordine lessicografico:



Dato che possiamo scrivere una biezione tra $\Sigma^\star$ e $\mathbb{N}$ abbiamo che i due insiemi hanno la stessa cardinalità

Dato che le stringhe che sono codifiche valide di MdT sono un sottoinsieme (infinito) di $\Sigma^\star$, allora anche l'insieme di tutte le MdT è numerabile

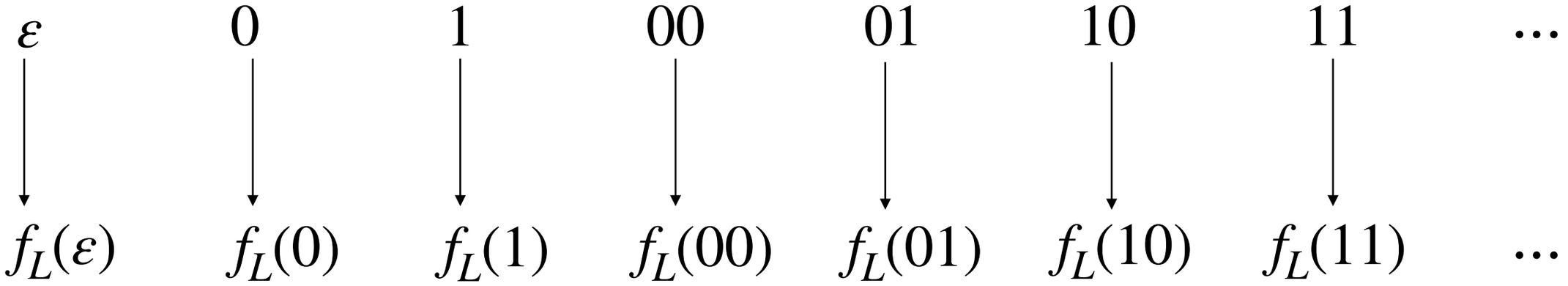
I linguaggi non sono numerabili

Dato un linguaggio $L \subseteq \Sigma^\star$ possiamo scrivere la funzione $f_L : \Sigma^\star \to \{0,1\}$ definita come

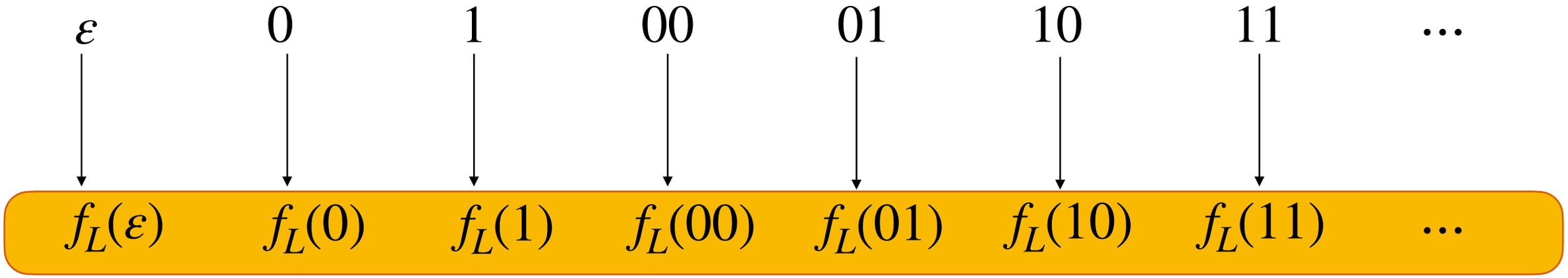
$$f_L(w) = \begin{cases} 1 & \text{se } w \in L \\ 0 & \text{se } w \notin L \end{cases}$$

Ovvero la funzione caratteristica dell'insieme L

Ogni funzione caratteristica può essere usata per costruire una sequenza infinita di zeri e uni:



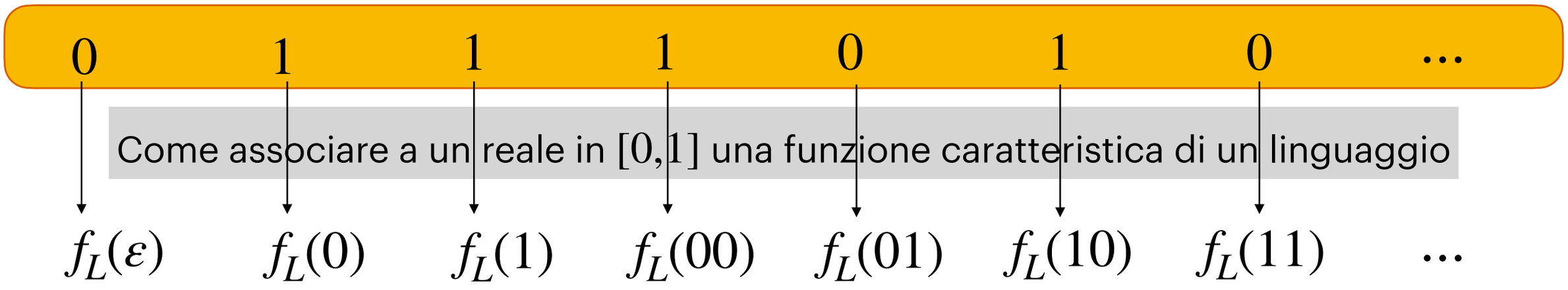
I linguaggi non sono numerabili



Espansione in base 2 della parte dopo la virgola di un numero reale tra 0 e 1

Quindi ogni linguaggio ha associata una espansione in base 2 della parte dopo la virgola di un reale tra 0 e 1

E, dato un reale tra 0 e 1 possiamo creare una funzione caratteristica. Per esempio:



I linguaggi non sono numerabili

Ricordando:

Che $[0,1] \subset \mathbb{R}$ ha la stessa cardinalità di $\mathbb{R}$

Che esiste una biezione tra linguaggi e $[0,1]$ (e quindi la cardinalità dei linguaggi è pari a quella dei reali)

Che l'insieme delle MdT ha la stessa cardinalità di $\mathbb{N}$

E che $\text{card}(\mathbb{N}) < \text{card}(\mathbb{R})$

Concludiamo che esistono linguaggi non riconoscibili da MdT (intuitivamente *"non ci sono abbastanza MdT per tutti i linguaggi"*)