UNIVERSITÀ
DEGLI STUDI DI TRIESTE

Computabilità, Complessità e Logica

Prof. Adriano Peron

Computabilità: Decidibilità

Macchina di Turing

- **Origine storica.**
- **Nel 1900 in un celebre intervento a un congresso il matematico David Hilbert elenca 23 problemi matematici che a suo parere costituiscono una sfida per la matematica del XX secolo.**
- **Il decimo problema riguarda la possibilità di trovare una procedura che risponda (test si/no) sull'esistenza di soluzioni intere per un generico polinomio a coefficienti interi.**
- "a process according to which it can be determined by a finite number of operations"
- **Dimostrare l'impossibilità di definire una tale procedura richiede la definizione precisa di un concetto di computazione e di computabilità o se preferiamo di ALGORITMO.**

Macchina di Turing

- **Origine storica.**
- Nel 1936 vengono proposte due definizioni
  - Alonzo Church propone il λ-calculus
  - Turing propone la definizione di una Macchina
- **E' stato dimostrato che entrambe le definizioni sono equivalenti**
- La possibilità di collegare la nozione di qualitativa di algoritmo con le definizioni di Turing e Church è stabilita dalla **Tesi di Church-Turing**
- **Una funzione sui numeri naturali può essere calcolata con un metodo effettivo se e solo è computabile da una macchina di Turing**.

Decidibilità

- Nelle lezioni precedenti si è introdotta la classe degli linguaggi che possono essere decisi da una macchia di Turing:
  - Per ogni parola del linguaggio esiste una computazione accettante
  - Per ogni parola non appartenete al linguaggio esiste una computazione di rifiuto
  - (in entrambi i casi la macchina termina).
- Verranno presentati a seguire esempi di problemi decidibili.
- Un problema di decisione verrà formulato come linguaggio
- Esempio: Il problema dell'accettazione di una parola w per un automa deterministico A
- E' decidibile il linguaggio

$$L = \{< A, w >: w \in \Sigma^*, A \text{ automa regolare deterministico accetta w}\}?$$

- $< A, w >$ è una stringa di caratteri che codifica l'automa A e w

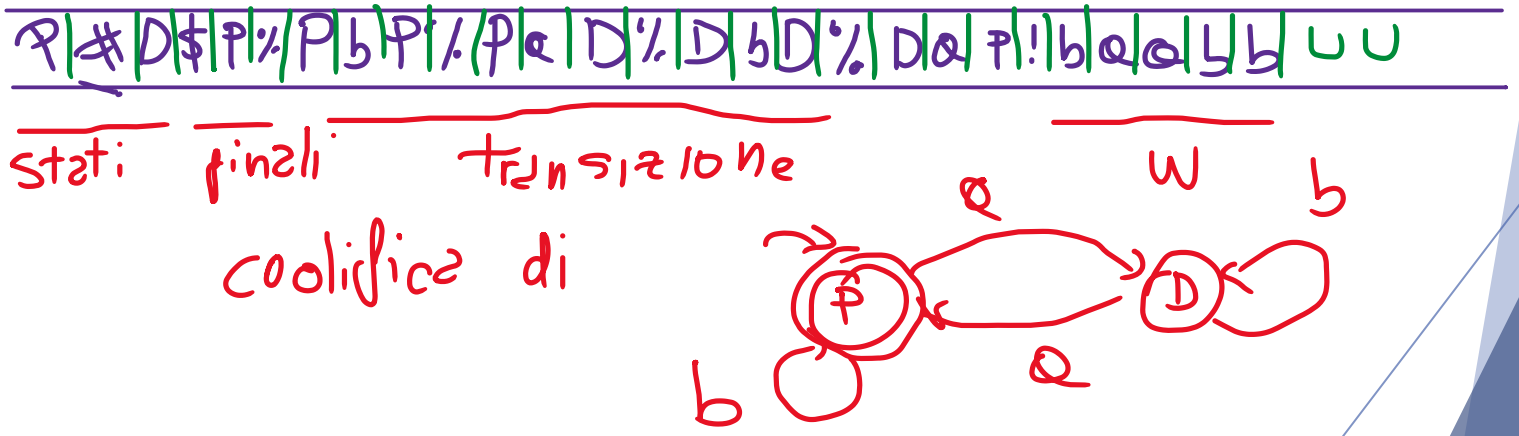
Decidibilità dell'accettazione per automi deterministici a stati finiti

L= $\{< A, w >: w \in \Sigma^*, A$ automa regolare deterministico accetta w$\}$

È decidibile.

Idea.

- **E' possibile scrivere una MdT che avendo $< A, w >$ sul nastro di ingresso simula l'esecusione di A sulla parola b**
- **Se la simulazione termina in stato di accettazione la MdT accetta**
- **Se la la simulazione termina in stato di non accettazione la MdT rifiuta.**



Decidibilità del vuoto per automi deterministici a stati finiti

L= $\{< A >:$ A automa regolare deterministico, L(A) $= \emptyset\}$

È decidibile.

Idea.

- **E' possibile scrivere una MdT che avendo $< A >$ sul nastro di ingresso usa la funzione di transizione per marcare gli stati raggiungibili dallo stato iniziale**
  - **Si marca lo stato iniziale**
  - **Iterativamente se lo stato q è marcato e (q,a,q') è una transizione si marca anche q'**
  - **Si termina quando non è più possibile aggiungere nuovi stati marcati**
- **Se dopo la marcatura degli stati raggiungibili uno stato finale è stato marcato la MdT termina con accettazione**
- **Se nessun stato finale è stato marcato la MdT termina con rifiuto.**

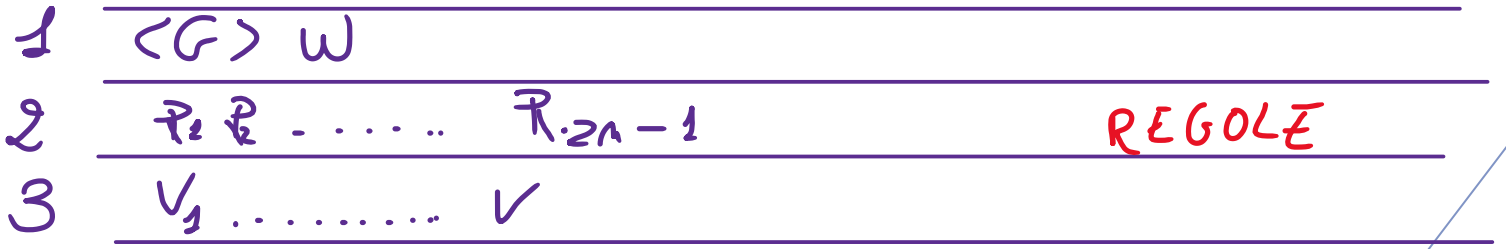
Decidibilità dell'accettazione per grammatiche contex free

L= $\{< G, w >: w \in \Sigma^*, G$ *grammatica context free*, G accetta w$\}$

È decidibile.

Idea.

- Assumiamo per semplicità la forma normale di Chomsky
- Ogni parola generabile è generabile in al più $2n-1$ riscritture con n la lunghezza di w
- E' possibile scrivere una MdT che usa tre nastri
  - Nel primo nastro ha la codifica $< G, w >$
  - Nel secondo nastro una codifica della scelta di $2n-1$ regole
  - Nel terzo nastro la derivazione correntemente in uso



Decidibilità dell'accettazione per grammatiche contex free

- **Ad ogni iterazione:**
  - **A partire dal simbolo iniziale si riscrive nel terzo nastro la parola corrente usando le regole elencate nel secondo nastro**
  - **Si confronta il risultato della riscrittura con w (nel primo nastro)**
  - **Se la parola è uguale si accetta e si termina**
  - **Se la parola è diversa si scrive nel secondo nastro la prossima (se esiste) sequenza di regola di lunghezza $2n-1$ da provare (le sequenze possono essere ordinate lessicograficamente) e si inzia una nuova iterazione**
  - **Se non esistono più sequenze di lunghezza $2n-1$ da provare si termina con rifiuto**

Decidibilità del vuoto per grammatiche context free

L= $\{< G >:$ G una grammatica contex free, L$(G) = \emptyset\}$

È decidibile.

Idea.

- La MdT ha $< G >$ sul nastro di ingresso.
- La MdT inizialmente marca tutti i simboli terminali.
- La MdT esegue iterativamente la seguente procedura:
  - Per ogni regola $V \longrightarrow V_1 \dots V_k$ se tulle le variabili $V_1 \dots V_k$ sono marcate si marca anche la variabile V
  - Se qualche variabile è stata marcata si itera la procedura altrimenti si termina la marcatura
- Se dopo la marcatura delle variabili la variabile iniziale non è stata marcata si termina con accettazione
- Se la variabile iniziale è stata marcata si termina con rifiuto.

Verso l'indecidibilità: il problema della terminazione

- Un esempio importante di linguaggio non decidibile è il problema dell'accettazione per una MdT.

$$L = \{< M, w >: w \in \Sigma^*, M \text{ una } Md\Gamma, \text{ M accetta w}\}$$

È indecidibile

- Il linguaggio L può essere riconosciuto (non deciso) da una MdT.
- Per provare che può essere riconosciuto si usa una MdT Universale (UMdT), una MdT in grado di simulare ogni altra MdT.
- La UMdT
  - Ha la codifica di una MdT M sul nastro insieme a una parola w sul nastro all'inizio della computazione
  - La UMdT simula il comportamento di M su w
  - La Umdt accetta se M accetta su w
  - La UMdT rifiuta se M rifiuta su w
  - La UMdT non termina se M non termina

Linguaggio L_{AMdT} Problema dell'accettazione di una MdT

- **TEOREMA. Il linguaggio**

$$L_{AMdT} = \{< M, w >: w \in \Sigma^*, M \text{ una } Md\Gamma, \text{ M accetta w}\}$$

è indecidibile

- **Prova.**
- **Per assurdo si assuma che esista una MdT H che:**
  - **Termina con accettazione se $< M, w >$ appartiene a L**
  - **Termina con rifiuto se $< M, w >$ non appartiene a L**
- **Si prenda ora la MdT D che usa H come una procedura e che la invoca su degli input speciali:**
  - **Termina con rifiuto su $< M >$ se H termina con accettazione su $< M >, < M >$**
  - **D termina con rifiuto su $< M >$ se la macchina M accetta la parola <M>**
  - **Termina con accettazione su $< M >$ se H termina con rifiuto su $< M >, < M >$**
  - **D termina con accettazione su $< M >$ se la macchina M la parola <M>**
- **Eseguiamo ora la macchina D sulla stringa $< D >$**
  - **D termina con rifiuto su $< D >$ se la macchina D accetta la parola <D>**
  - **D termina con accettazione su $< D >$ se la macchina D rifiuta la parola <D>**
- **Assurdo!**

Problema dell'accettazione di una MdT

- **La prova utilizza una tecnica di diagonalizzazione**

Input

MdT

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$
M_1	*accept*		*accept*		
M_2	*accept*	*accept*	*accept*	*accept*	
M_3					$\cdots$
M_4	*accept*	*accept*			
$\vdots$			$\vdots$		

M_2 accetta $\langle M_4 \rangle$
M_1 rifiuta o cicla su $\langle M_4 \rangle$

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$
M_1	*accept*	*reject*	*accept*	*reject*	
M_2	*accept*	*accept*	*accept*	*accept*	
M_3	*reject*	*reject*	*reject*	*reject*	$\cdots$
M_4	*accept*	*accept*	*reject*	*reject*	
$\vdots$			$\vdots$		

Se MdT H permette di completare la tabella

Problema dell'accettazione di una MdT

- **La prova utilizza una tecnica di diagonalizzazione**

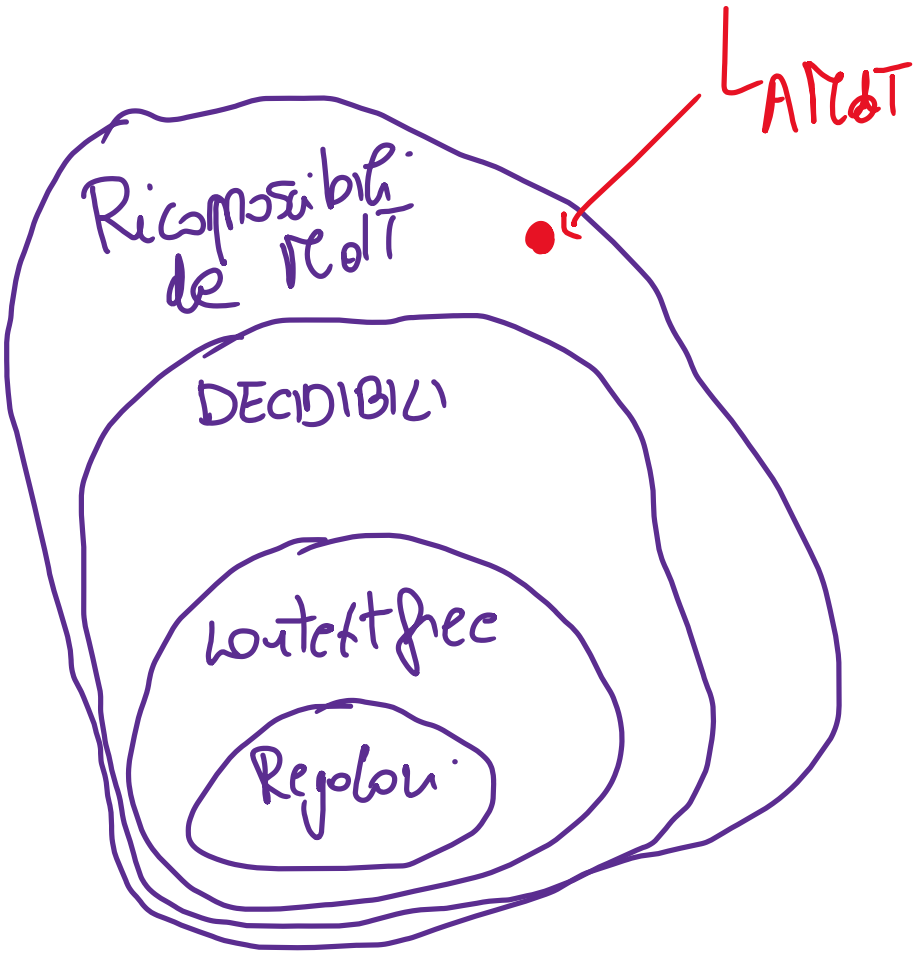
	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$
M_1	*accept*	*reject*	*accept*	*reject*	
M_2	*accept*	*accept*	*accept*	*accept*	$\cdots$
M_3	*reject*	*reject*	*reject*	*reject*	
M_4	*accept*	*accept*	*reject*	*reject*	
$\vdots$			$\vdots$		

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$	$\langle D \rangle$	$\cdots$
M_1	*accept*	*reject*	*accept*	*reject*		*accept*	
M_2	*accept*	*accept*	*accept*	*accept*	$\cdots$	*accept*	$\cdots$
M_3	*reject*	*reject*	*reject*	*reject*		*reject*	
M_4	*accept*	*accept*	*reject*	*reject*		*accept*	
$\vdots$			$\vdots$		$\ddots$		
D	*reject*	*reject*	*accept*	*accept*		?	
$\vdots$			$\vdots$				$\ddots$

D complementa gli elementi diagonali

Deve complementare se stesso!

Gerarchia delle classi di linguaggi?



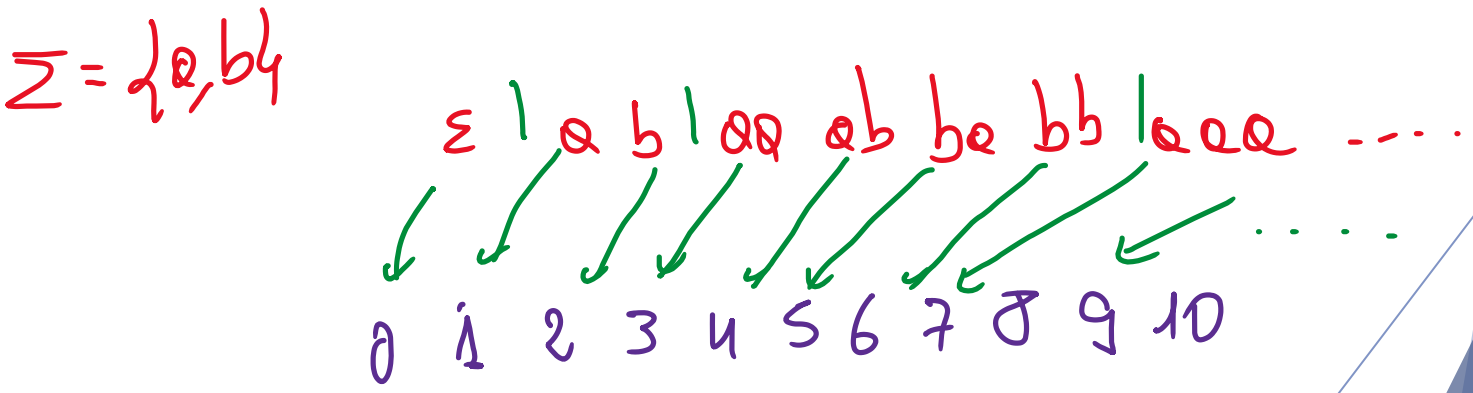
Esistono linguaggi non riconoscibili da MdT?

Linguaggi non riconoscibili da MdT

- **L'esistenza di linguaggi non riconoscibile da MdT può essere provata considerando la cardinalità dell'insieme delle MdT che riconoscono linguaggi su Σ**

Proprietà

- **L'insieme delle parole su Σ (cioè Σ^*) è enumerabile.**
- **Assumiamo un ordinamento totale dei simboli in Σ**
- **Ordiniamo in modo lessicografico le parole di Σ^***
- **Se w_i è la i-esima parola dell'ordinamento, w_i è in corrispondenza con i**



Numerabilità dell'insieme dei linguaggi riconosciuti da MdT

Proprietà

- **L'insieme delle MdT su è enumerabile.**
- **Una MdT è una struttura finita**
- **Una MdT può essere codificata mediante una stringa finita usando un opportuno alfabeto alfabeto Σ.**
- **Ogni MdT può essere codificata rispetto a un comune alfabeto Σ.**
- **Le codifiche di tutte le possibili MdT su Σ sono un sottoinsieme di Σ^***
- **Poiché Σ^* è numerabile le possibili MdT sono numerabili.**

Conseguenza

L'insieme dei linguaggi riconosciuti da una MdT è numerabile!

L'insieme dei linguaggi su un alfabeto Σ non è numerabile

Prova.

- **Costruiamo una rappresentazione efficace di un linguaggio su Σ**
- **Ordiniamo lessicograficamente le parole di Σ^* (w_i è *la i − esima parola dell'ordinamento*)**
- **Stringa caratteristica di un linguaggio.**
- **Una sequenza infinita α di valori binari (0,1)**
- **α(i) = 1 se w_i appartiene al linguaggio**
- **α(i) = 0 se w_i *non* appartiene al linguaggio**

$\Sigma = \{0,1\}$

$\varepsilon \mid 0 \mid 1 \mid 00 \mid 01 \mid 10 \mid 11 \mid 000 \mid 001 \mid \ldots \ldots$
0 1 2 3 4 5 6 7 8

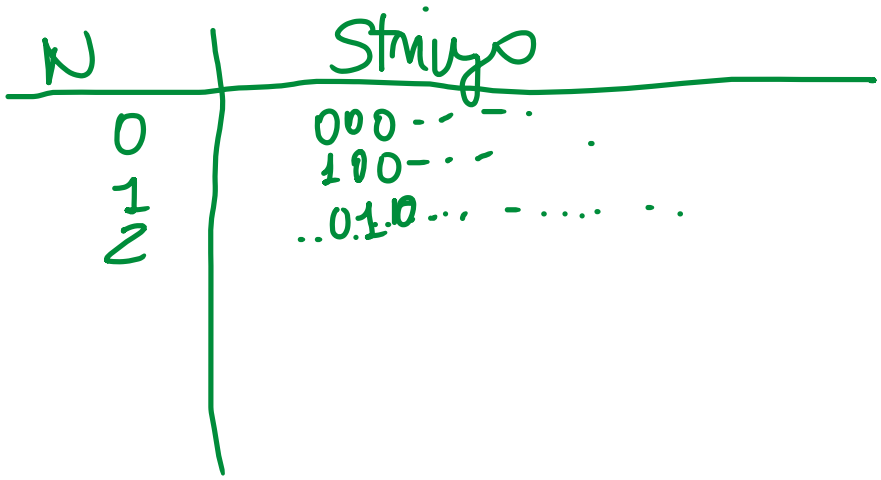
$L = \{\varepsilon, 0, 00, 000, \ldots$

STRINGA CARATTERISTICA : 110100010....

L'insieme dei linguaggi su un alfabeto Σ non è numerabile

Prova.

- **Un linguaggio è associato in modo univoco alla sua stringa caratteristica.**
- **Le stringhe caratteristiche dei linguaggi sono numerabili?**
- **No!**
- **Lo si può provare con una costruzione diagonale.**
- **Si assuma per assurdo che le stringhe caratteristiche siano numerabili**



Teorema. L'insieme dei linguaggi su un alfabeto Σ non è numerabile

Prova.

- **Costruiamo una stringa binaria α con le seguenti caratteristiche**
- **$\alpha(i)$ ha il complemento dell'i-esimo valore della i-esima stringa**
- **Conseguenza.**
- **α differisce in almeno una posizione con ogni stringa (la i-esima con la i-esima stringa)**
- **α non è compresa nell'enumerazione delle stringhe**
- **L'ipotesi dell'enumerabilità delle stringhe ha portato ad un assurdo.**
- **L'insieme delle stringhe caratteristiche non è enumerabile.**
- L'insieme dei linguaggi su un alfabeto Σ non è numerabile

Corollario. Esistono linguaggi non riconoscibili da MdT

Prova.

- **Come visto l'insieme dei linguaggi su Σ fissato non è numerabile**
- **L'insieme delle macchine definibili usando l'alfabeto di simboli in Σ è numerabile**
- **Sono numerabili i linguaggi riconosciuti da MdT su un Σ**
- **Esistono linguaggi su Σ non riconosciuti.**

Proprietà dei Linguaggi decidibili

Teorema. Un linguaggio L è decidibile se e solo L e $\bar{L}$ sono riconoscibili da MdT.

Prova.

- Sia L decidibile.
- Se L è decidibile è riconoscibile da MdT (deterministica).
- Se L è decidibile esiste una MdT M che termina con rifiuto per ogni parola $w \notin L$
- Sia M' una MdT costruita a partire da M che complementa gli stati di accept e reject.
- M' riconosce $\bar{L}$

Applicazione della proprietà dei Linguaggi decidibili

Si ricorda che $L_{AMdT} = \{< M, w >: w \in \Sigma^*, M \text{ una MdT}, \text{M accetta w}\}$

è indecidibile

Sappiamo che L_{AMdT} è riconosciuto da MdT (MdT Universale).

Per la propreità dei linguaggi decidibili il linguaggio complemento di L_{AMdT} non può essere riconosciuto da MdT.

In generale.

Il linguaggio complemento di un linguaggio riconoscibile da MdT ma non decidibile non è riconosciuto da MdT.

Proprietà dei Linguaggi decidibili

Teorema. Un linguaggio L è decidibile se e solo L e $\bar{L}$ sono riconoscibili da MdT.

Prova.

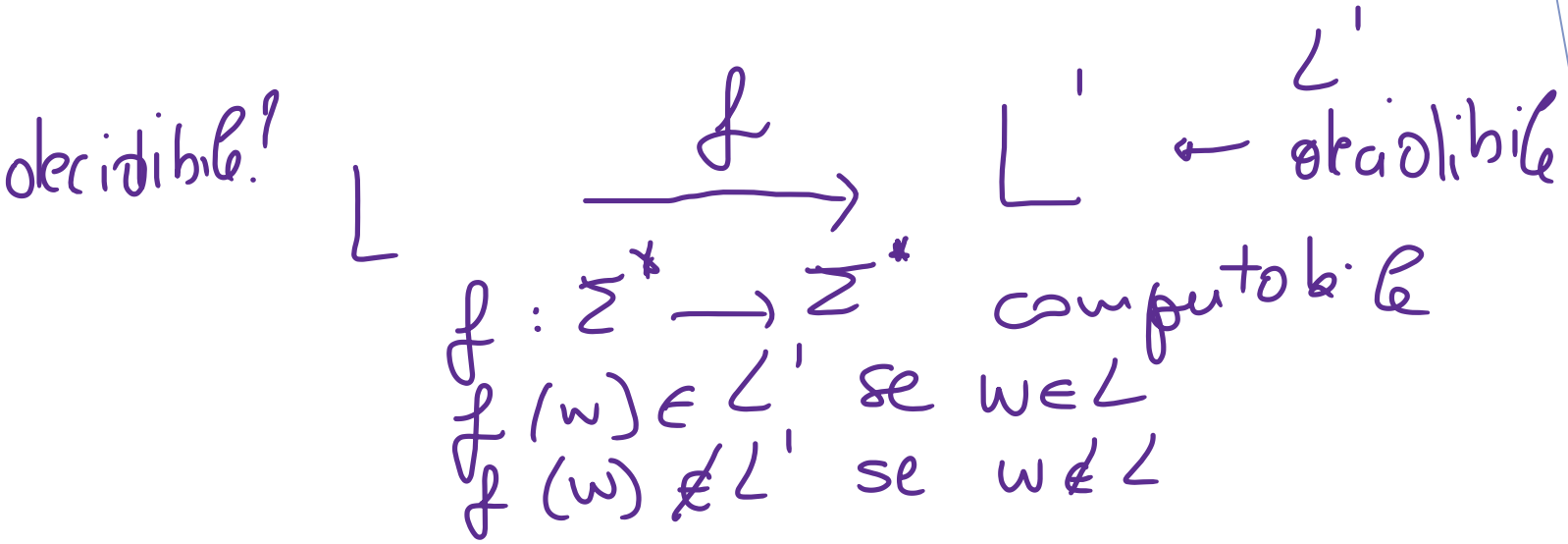
- Siano L e $\bar{L}$ riconoscibili da MdT.
- Sia M la MdT che riconosce L e M' la MdT che riconosce $\bar{L}$
- E' possibile costruire una MdT M'' che simula una computazione in parallelo di M e M'
- ad esempio.
  - si prenda una MdT a tre nastri, il primo per l'input, il secondo per la computazione di M e il terzo per la computazione di M'.
  - Si simuli l'avanzamento parallelo di M e di M' alternando un passo di M con un passo di M'
  - Se per prima arriva in accettazione M, M'' termina con accept
  - Se per prima arriva in accettazione M', M'' termina con reject

Risolvere problemi di decidibilità/indecidibilità

- **Una tecnica diffusa per risolvere la decidibilità/indecidibilità di un problema P è la tecnica di riduzione a un problema P' già noto.**
- **Il problema P viene trasformato mediante una funzione computabile nel problema P' applicando poi quanto noto sul problema P'**
- **Funzione computabile: Una funzione $f: \Sigma^* \to \Sigma^*$ è computabile se esiste una MdT che per ogni input $w \in \Sigma^*$ termina riportando sul nastro $f(w)$.**

Risolvere problemi di decidibilità

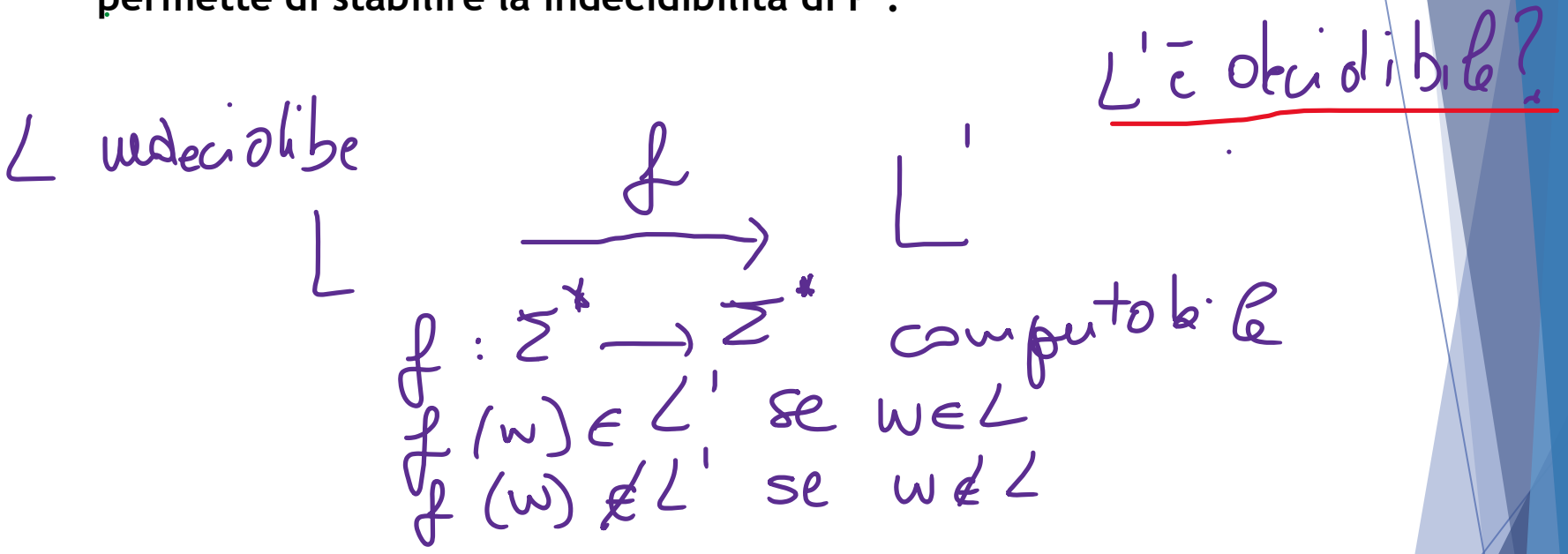
- **Riduzione di un problema P a un problema decidibile P': permette di stabilire la decidibilità di P.**



- **Per la decidibilità uso in cascata la MdT per la funzione computabile**
- **Applico poi la MdT che permette di decidere L' su f(w)**

Risolvere problemi di indecidibilità

- **Riduzione di un problema indecidibile P a un problema P':**
  permette di stabilire la indecidibilità di P'.



- **La riduzione di un problema indecidibile a un problema non noto permette di dimostrare la indecidibilità del problema non noto**
- **Per assurdo la decidibilità di P' implicherebbe la decidibilità di P per quanto già visto.**

Esempi di riduzione: il problema della terminazione di una MdT

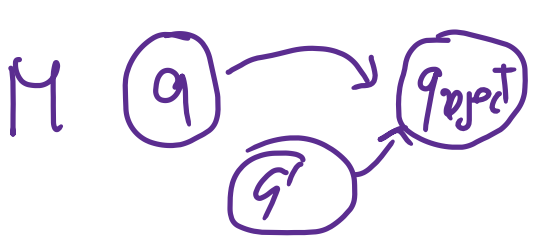
- **Problema della terminazione:** data una MdT M e una parola w è decidibile se M termina (in accettazione o rifiuto)?
- $L_{HALT} = \{< M > w : w \in \Sigma^*, M \ termina \ su \ input \ w\}$

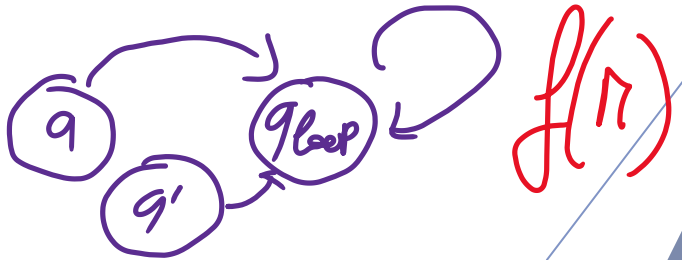
TEOREMA: L_{HALT} è indecidibile.

Prova per riduzione del problema dell'accettazione delle MdT (linguaggio L_{AMdT}) al problema della terminazione.

Riduzione f: si trasforma la MdT M rimuovendo lo stato di reject e sostituendolo con un ciclo non terminante.

La MdT M' risultante termina su w se e solo se M accetta w. ($< M > w \in L_{AMdT}$ iff $(f(< M >)w \in L_{HALT})$





Esempi di riduzione: il problema del vuoto delle MdT

- **Problema del vuoto:** data una MdT M, L(M)= ϕ?

$$L_{EMdT} = \{< M >: \text{L(M)= } \phi\}$$

TEOREMA: L_{EMdT} **è indecidibile.**

Prova per riduzione del problema dell'accettazione delle MdT (linguaggio L_{AMdT}**) al problema del vuoto.**

Riduzione f: dato l'input <M>w si trasforma la MdT M in una macchina M' (f(<M>,w)=M') in modo che rifiuti tutte le parole diverse dà w e si comporti come M su w.

M accetta w se e solo se L(f(<M>,w)) $\neq \emptyset$.

Se il problema del vuoto fosse decidibile L_{AMdT} **sarebbe decidibile.**

Esempi di riduzione: il problema dell'equivalenza delle MdT

- **Problema dell'equivalenza :** date due MdT M e M', L(M)=L(M')?

$$L_{EQUIV} = \{< M >< M' >: L(M) = L(M')\}$$

TEOREMA: L_{EQUIV} **è indecidibile.**

Prova per riduzione del problema del vuoto delle MdT (linguaggio L_{EMdT}) al problema dell'equivalenza.

Riduzione f: dato l'input <M> si crea una MdT $< M_E >$

che non accetta nessun input ($L(M_E) = \emptyset$).

$f(<M>) = <M><M_E>$

$L(M) = \emptyset$ se e solo se $f(<M>) \in L_{EQUIV}$

Se il problema dell'equivalenza fosse decidibile L_{EMdT} sarebbe decidibile.