# Lezione di Informatica Teorica: Teorema di Rice

Appunti da Trascrizione Automatica

30 giugno 2025

## Indice

# 1 Introduzione al Teorema di Rice

Abbiamo precedentemente studiato linguaggi come $L_e = \{\langle M \rangle \mid L(M) = \emptyset\}$ (il linguaggio delle macchine di Turing il cui linguaggio è vuoto) e $L_{ne} = \{\langle M \rangle \mid L(M) \neq \emptyset\}$ (il linguaggio delle macchine di Turing il cui linguaggio non è vuoto). Abbiamo dimostrato che questi linguaggi sono indecidibili. Tuttavia, la loro indecidibilità non è un caso isolato, ma rientra in un risultato molto più generale: il Teorema di Rice. Questo teorema si applica a linguaggi che contengono le codifiche di macchine di Turing le cui proprietà soddisfano determinati criteri.

## 1.1 Definizione di Proprietà di Macchine di Turing

Intuitivamente, una macchina di Turing ha una certa proprietà se possiede determinate caratteristiche. Per formalizzare questo concetto:

**Definizione 1** (Proprietà di Macchine di Turing). *Una **proprietà** $P$ di macchine di Turing è un insieme di codifiche di macchine di Turing. Una macchina di Turing $M$ si dice che ha la proprietà $P$ se e solo se la sua codifica $\langle M \rangle$ appartiene a $P$.*

Associato a una proprietà $P$, definiamo il **linguaggio della proprietà** $P$ come:

$$L_P = \{\langle M \rangle \mid M \text{ ha la proprietà } P\} = \{\langle M \rangle \mid \langle M \rangle \in P\}$$

In pratica, $L_P$ è semplicemente la proprietà $P$ stessa quando vista come un linguaggio.

## 1.2 Classificazione delle Proprietà

Le proprietà delle macchine di Turing possono essere categorizzate in due grandi famiglie:

1. **Proprietà Strutturali (o Sintattiche)**: Riguardano la struttura interna della macchina di Turing o il suo comportamento computazionale, ma non direttamente il linguaggio che essa riconosce.
2. **Proprietà Semantiche (o di Linguaggi)**: Riguardano esclusivamente il linguaggio riconosciuto dalla macchina di Turing, indipendentemente dalla sua implementazione specifica o dal suo comportamento interno (purché sia funzionalmente equivalente).

Il Teorema di Rice si applica specificamente alle proprietà semantiche.

**Definizione 2** (Proprietà Semantica). *Una proprietà $P$ di macchine di Turing è detta **semantica** se per ogni coppia di macchine di Turing $M_1$ e $M_2$:*

$$L(M_1) = L(M_2) \implies (\langle M_1 \rangle \in P \iff \langle M_2 \rangle \in P)$$

*Ciò significa che l'appartenenza di una macchina $M$ a una proprietà semantica $P$ dipende unicamente dal linguaggio $L(M)$ che essa riconosce. Se due macchine riconoscono lo stesso linguaggio, o entrambe possiedono la proprietà $P$ o nessuna delle due la possiede. Per questa ragione, le proprietà semantiche sono anche chiamate **proprietà di linguaggi**.*

**Esempio 1** (Proprietà Strutturale). *Sia $P_1$ la proprietà: la macchina di Turing $M$ ha esattamente 5 stati. $P_1 = \{\langle M \rangle \mid M \text{ ha 5 stati}\}$. Questa è una proprietà strutturale. Per dimostrare che non è semantica, troviamo un controesempio: Siano $M_1$ e $M_2$ due macchine di Turing. $M_1$ ha 5 stati e riconosce un linguaggio $L$. $M_2$ è costruita da $M_1$ aggiungendo uno stato irraggiungibile (o una transizione superflua, ecc.), cosicché $M_2$ abbia 6 stati ma riconosca lo stesso linguaggio $L$. In questo caso, $L(M_1) = L(M_2)$, ma $\langle M_1 \rangle \in P_1$ mentre $\langle M_2 \rangle \notin P_1$. Dunque $P_1$ non è semantica.*

**Esempio 2** (Proprietà Semantica)**.** *Sia $P_2$ la proprietà: il linguaggio riconosciuto da M contiene solo stringhe di lunghezza pari.* $P_2 = \{\langle M \rangle \mid \forall s \in L(M), |s| \text{ è pari}\}$. *Questa è una proprietà semantica. Se* $L(M_1) = L(M_2)$, *allora o tutte le stringhe di* $L(M_1)$ *(e quindi di* $L(M_2)$*) hanno lunghezza pari, o nessuna (o alcune) ce l'hanno. L'appartenenza a* $P_2$ *dipende solo dal linguaggio.*

**Esempio 3** (Proprietà Semantica: $L_e$)**.** *Il linguaggio* $L_e = \{\langle M \rangle \mid L(M) = \emptyset\}$ *è l'insieme delle macchine di Turing il cui linguaggio è vuoto. Questo è un esempio di proprietà semantica, poiché dipende solo dal linguaggio riconosciuto (in questo caso, il linguaggio vuoto).*

### 1.3 Proprietà Banali

**Definizione 3** (Proprietà Banale)**.** *Una proprietà P è detta **banale** se:*

1. *Non contiene alcuna macchina di Turing:* $P = \emptyset$.
2. *Contiene tutte le macchine di Turing:* $P = \{\langle M \rangle \mid M \text{ è una macchina di Turing}\}$.

*Se P è una proprietà di linguaggi (cioè semantica), allora P è banale se e solo se* $P = \emptyset$ *(nessun linguaggio ha la proprietà) o* $P = RE$ *(tutti i linguaggi ricorsivamente enumerabili hanno la proprietà).*

Le proprietà banali sono sempre **decidibili**. Se una proprietà è banale nel senso che $P = \emptyset$, possiamo sempre rispondere "no" per qualsiasi macchina data. Se $P$ contiene tutte le macchine, possiamo sempre rispondere "sì". Il problema di decidere se una macchina possiede una proprietà banale è quindi banale esso stesso.

## 2 Teorema di Rice

Il Teorema di Rice è un risultato fondamentale nella teoria della computabilità, che generalizza l'indecidibilità di problemi come $L_e$ e $L_{ne}$.

**Teorema 1** (Teorema di Rice)**.** *Ogni proprietà non banale dei linguaggi ricorsivamente enumerabili (RE) è indecidibile. In altre parole, se P è una proprietà semantica (proprietà di linguaggi) tale che* $P \neq \emptyset$ *e* $P \neq RE$, *allora il linguaggio* $L_P = \{\langle M \rangle \mid L(M) \in P\}$ *è indecidibile.*

### 2.1 Dimostrazione del Teorema di Rice

Sia $P$ una proprietà semantica non banale dei linguaggi RE. Vogliamo dimostrare che $L_P$ è indecidibile. Procediamo con una riduzione dal Problema di Halting Universale, $L_u = \{\langle M, w \rangle \mid M \text{ accetta } w\}$, che sappiamo essere indecidibile.

La dimostrazione si divide in due casi, a seconda che il linguaggio vuoto $\emptyset$ appartenga o meno alla proprietà $P$.

#### 2.1.1 Caso 1: Il linguaggio vuoto non appartiene a $P$ ($\emptyset \notin P$)

Poiché $P$ è una proprietà non banale, e $\emptyset \notin P$, deve esistere almeno un linguaggio $L \in P$ tale che $L \neq \emptyset$. (Se tutti i linguaggi in $P$ fossero vuoti, e $\emptyset \notin P$, allora $P$ sarebbe $\emptyset$, contraddicendo l'ipotesi che $P$ sia non banale). Dato che $L \in RE$, deve esistere una macchina di Turing $M_L$ tale che $L(M_L) = L$.

Costruiamo una macchina di Turing $N$ a partire da una coppia $\langle M, w \rangle$ (input per $L_u$) e da $M_L$. La macchina $N$ è una nuova macchina (la cui codifica è l'output della nostra riduzione) che prende un input $x$. La sua logica di funzionamento è la seguente:

**Costruzione della macchina $N_{M,w}$ (che chiamiamo $N$ per semplicità):**

1. Su input $x$:
2. Ignora l'input $x$ e simula $M$ sull'input $w$.
3. Se la simulazione di $M$ su $w$ accetta:
   (a) Inizia a simulare $M_L$ sull'input $x$.
   (b) Se $M_L$ accetta $x$, allora $N$ accetta $x$.
   (c) Se $M_L$ rifiuta $x$, allora $N$ rifiuta $x$.
4. Se la simulazione di $M$ su $w$ non accetta (ovvero, rifiuta o loopa):
   (a) $N$ non accetta (rifiuta o loopa).

Ora analizziamo il comportamento di $N$ per dimostrare che la riduzione funziona:

**i) Se $\langle M, w \rangle \in L_u$ (cioè, $M$ accetta $w$):** In questo caso, la simulazione di $M$ su $w$ al passo 2 della costruzione di $N$ terminerà e accetterà. Di conseguenza, $N$ procederà sempre al passo 3 e simulerà $M_L$ su $x$. Questo significa che $N$ accetta $x$ se e solo se $M_L$ accetta $x$. Quindi, $L(N) = L(M_L) = L$. Poiché abbiamo stabilito che $L \in P$, ne consegue che $\langle N \rangle \in L_P$.

**ii) Se $\langle M, w \rangle \notin L_u$ (cioè, $M$ non accetta $w$):** In questo caso, la simulazione di $M$ su $w$ al passo 2 della costruzione di $N$ non terminerà accettando (o rifiuterà, o loopa). Di conseguenza, $N$ non raggiungerà mai il passo 3 e quindi non accetterà mai alcun input $x$. Quindi, $L(N) = \emptyset$. Poiché abbiamo assunto $\emptyset \notin P$, ne consegue che $\langle N \rangle \notin L_P$.

Questa costruzione definisce una funzione calcolabile $f : \langle M, w \rangle \mapsto \langle N \rangle$ tale che $\langle M, w \rangle \in L_u \iff \langle N \rangle \in L_P$. Poiché $L_u$ è indecidibile e $L_u \leq_m L_P$, concludiamo che $L_P$ è indecidibile.

### 2.1.2 Caso 2: Il linguaggio vuoto appartiene a $P$ ($\emptyset \in P$)

Se $\emptyset \in P$, consideriamo la proprietà $\overline{P}$, definita come il complemento di $P$ rispetto all'insieme di tutti i linguaggi RE: $\overline{P} = RE \setminus P$.

- $\overline{P}$ è una proprietà semantica, poiché se $P$ lo è, anche il suo complemento lo è.
- Poiché $\emptyset \in P$, ne consegue che $\emptyset \notin \overline{P}$.
- Se $P$ è non banale, allora $\overline{P}$ è anch'essa non banale. (Se $P = \emptyset$, $\overline{P} = RE$. Se $P = RE$, $\overline{P} = \emptyset$. In entrambi i casi, $\overline{P}$ sarebbe banale. Ma abbiamo assunto che $P$ è non banale, quindi anche $\overline{P}$ è non banale).

Riassumendo, $\overline{P}$ è una proprietà semantica non banale e $\emptyset \notin \overline{P}$. Questo è esattamente il Caso 1 che abbiamo appena dimostrato. Quindi, il linguaggio $L_{\overline{P}} = \{\langle M \rangle \mid L(M) \in \overline{P}\}$ è indecidibile.

Ora, supponiamo per assurdo che $L_P$ sia decidibile. Allora esisterebbe una macchina di Turing decisore $D_P$ che decide $L_P$. Utilizzando $D_P$, potremmo costruire una macchina di Turing $D_{\overline{P}}$ che decide $L_{\overline{P}}$ nel modo seguente: **Costruzione di $D_{\overline{P}}$ su input $\langle M \rangle$:**

1. Esegui $D_P$ su $\langle M \rangle$.

2. Se $D_P$ accetta $\langle M \rangle$, allora $D_{\overline{P}}$ rifiuta $\langle M \rangle$.

3. Se $D_P$ rifiuta $\langle M \rangle$, allora $D_{\overline{P}}$ accetta $\langle M \rangle$.

Questa macchina $D_{\overline{P}}$ deciderebbe $L_{\overline{P}}$. Tuttavia, abbiamo appena dimostrato che $L_{\overline{P}}$ è indecidibile. Abbiamo raggiunto una contraddizione. Pertanto, la nostra supposizione iniziale che $L_P$ fosse decidibile deve essere falsa. Concludiamo quindi che $L_P$ è indecidibile anche nel Caso 2.

Combinando i due casi, il Teorema di Rice è dimostrato. Ogni proprietà non banale dei linguaggi RE è indecidibile.

# 3 Applicazioni del Teorema di Rice

Il Teorema di Rice fornisce un potente strumento per dimostrare l'indecidibilità di un'ampia classe di problemi. Per applicarlo, è sufficiente verificare che la proprietà in questione sia:

1. Una proprietà di linguaggi (cioè semantica).

2. Non banale.

Se entrambe le condizioni sono soddisfatte, allora il problema di decidere se una macchina di Turing possiede tale proprietà è indecidibile.

**Esempio 4** (Indecidibilità di $L_e$ e $L_{ne}$). *1.* $L_e = \{\langle M \rangle \mid L(M) = \emptyset\}$*:*

- ***Proprietà semantica?****: Sì, dipende solo dal linguaggio* $L(M)$*.*
- ***Non banale?****: Sì. Contiene il linguaggio vuoto (quindi non è* $\emptyset$*). Non contiene, ad esempio,* $\Sigma^*$ *(quindi non è RE).*

*Dato che è semantica e non banale, per il Teorema di Rice,* $L_e$ *è indecidibile.*

*2.* $L_{ne} = \{\langle M \rangle \mid L(M) \neq \emptyset\}$*:*

- ***Proprietà semantica?****: Sì, dipende solo dal linguaggio* $L(M)$*.*
- ***Non banale?****: Sì. Contiene, ad esempio,* $\Sigma^*$ *(quindi non è* $\emptyset$*). Non contiene* $\emptyset$ *(quindi non è RE).*

*Dato che è semantica e non banale, per il Teorema di Rice,* $L_{ne}$ *è indecidibile.*

**Esempio 5** (Decidere se $L(M)$ è finito). *Sia* $L_{finito} = \{\langle M \rangle \mid L(M) \text{ è finito}\}$*.*

- ***Proprietà semantica?****: Sì, la finitezza di un linguaggio è una sua proprietà intrinseca.*
- ***Non banale?****: Sì. Contiene linguaggi finiti (e.g.,* $L(M) = \emptyset$ *o* $L(M) = \{a\}$*), quindi non è* $\emptyset$*. Non contiene linguaggi infiniti (e.g.,* $L(M) = \Sigma^*$*), quindi non è RE.*

*Per il Teorema di Rice,* $L_{finito}$ *è indecidibile.*

**Esempio 6** (Decidere se $L(M)$ è infinito). *Sia* $L_{infinito} = \{\langle M \rangle \mid L(M) \text{ è infinito}\}$*.*

- ***Proprietà semantica?****: Sì.*
- ***Non banale?****: Sì. Contiene linguaggi infiniti (e.g.,* $L(M) = \Sigma^*$*), quindi non è* $\emptyset$*. Non contiene linguaggi finiti (e.g.,* $L(M) = \emptyset$*), quindi non è RE.*

*Per il Teorema di Rice, $L_{infinito}$ è indecidibile.*

**Esempio 7** (Decidere se $L(M)$ è riconosciuto solo da macchine con 5 stati)**.** *Sia $P_{solo5stati} = \{\langle M \rangle \mid L(M)$ è riconosciuto solo da macchine con 5 stati$\}$.*

- ***Proprietà di linguaggi?***: *Sì, è una proprietà semantica.*
- ***Non banale?***: *No, è banale. Ogni linguaggio ricorsivamente enumerabile che può essere riconosciuto da una macchina di Turing con 5 stati, può essere riconosciuto anche da una macchina con più di 5 stati (basta aggiungere stati irraggiungibili). Quindi, nessun linguaggio può essere riconosciuto* solo *da macchine con 5 stati. Pertanto, $P_{solo5stati} = \emptyset$. Essendo $\emptyset$, è una proprietà banale.*

*Poiché è una proprietà banale, $P_{solo5stati}$ è **decidibile** (la risposta è sempre "no"). Il Teorema di Rice non si applica per dimostrare l'indecidibilità in questo caso.*

## 3.1 Altri Esempi specifici

Consideriamo il linguaggio $L = \{w\#A \mid w \in \{0,1\}^+, A = w \vee A = w^R\}$. Questo linguaggio è RE. Di seguito, una descrizione di una macchina di Turing a 2 nastri che riconosce $L$:

- Inizialmente, il nastro 1 contiene $w\#A$.
- La macchina copia $w$ sul nastro 2.
- Quando incontra '#', si sposta all'inizio di $w$ sul nastro 2.
- Non deterministicamente, la macchina può scegliere tra due rami:
  1. **Verifica** $A = w$: Confronta $A$ sul nastro 1 con $w$ sul nastro 2, leggendo entrambi da sinistra a destra. Se corrispondono e si arriva alla fine, accetta.
  2. **Verifica** $A = w^R$: Confronta $A$ sul nastro 1 da sinistra a destra con $w$ sul nastro 2 da destra a sinistra. Se corrispondono e si arriva alla fine, accetta.

**Esempio 8** (Decidere se $L(M) = L$)**.** *Sia $P_L = \{\langle M \rangle \mid L(M) = L\}$, dove L è il linguaggio definito sopra.*

- ***Proprietà di macchine?***: *Sì.*
- ***Proprietà semantica?***: *Sì, dipende solo dal fatto che il linguaggio riconosciuto sia esattamente L.*
- ***Non banale?***: *Sì. Contiene il linguaggio L (quindi non è $\emptyset$). Non contiene, ad esempio, il linguaggio $\Sigma^*$ (quindi non è RE).*

*Per il Teorema di Rice, $P_L$ è **indecidibile**.*

**Esempio 9** (Decidere se ogni stringa di $L(M)$ è accettata in al più 100 passi)**.** *Sia $P_{100steps} = \{\langle M \rangle \mid \forall s \in L(M), M$ accetta $s$ in $\leq 100$ passi$\}$.*

- ***Proprietà di macchine?***: *Sì.*
- ***Proprietà semantica?***: *No. Dipende dal comportamento computazionale (numero di passi). Controesempio: una macchina $M_1$ accetta "a" in 10 passi. Un'altra macchina $M_2$ potrebbe accettare "a" in 200 passi (es. loopando inutilmente prima di accettare). $L(M_1) = L(M_2) = \{a\}$. Ma $M_1 \in P_{100steps}$ e $M_2 \notin P_{100steps}$.*

- ***Banale?***: *No. Contiene macchine che accettano stringhe corte in pochi passi (e.g., una macchina che accetta solo $\epsilon$ in 5 passi). Non è RE (poiché non tutte le macchine soddisfano la proprietà).*

*Poiché $P_{100steps}$ non è semantica, il Teorema di Rice **non si applica**. Questo problema è in realtà **decidibile**. Una macchina per questo problema può simulare M su tutte le stringhe di lunghezza $\leq 100$ per 100 passi. Se M accetta una stringa più lunga di 100, o non accetta una stringa in $L(M)$ entro 100 passi, allora rifiuta. Altrimenti accetta.*

**Esempio 10** (Decidere se $M$ non accetta stringhe di $L$ di lunghezza 100)**.** *Sia $P_{no100} = \{\langle M \rangle \mid L(M) \cap \{s \mid |s| = 100\} = \emptyset\}$, dove L è il linguaggio definito in precedenza. In altre parole, M non accetta alcuna stringa del linguaggio L che abbia lunghezza 100.*

- ***Proprietà di macchine?***: *Sì.*
- ***Proprietà semantica?***: *Sì, dipende esclusivamente dal linguaggio $L(M)$ e dalla sua intersezione con l'insieme delle stringhe di lunghezza 100.*
- ***Non banale?***: *Sì.*
  - *Non è $\emptyset$: una macchina che accetta solo 0#0 (lunghezza 3) appartiene a $P_{no100}$, poiché non accetta alcuna stringa di lunghezza 100.*
  - *Non è RE: una macchina che accetta una stringa di L di lunghezza 100 (se esiste) non appartiene a $P_{no100}$. Se esistono stringhe di L di lunghezza 100, allora non tutte le macchine RE appartengono a $P_{no100}$.*

*Poiché $P_{no100}$ è semantica e non banale, per il Teorema di Rice, è **indecidibile**.*