Metodologia formale secondo Bresolin

🎯 OBIETTIVO GENERALE

Dimostrare che un linguaggio L è **indecidibile** usando **riduzioni** da problemi noti indecidibili.

📚 DEFINIZIONI FONDAMENTALI

Linguaggio Indecidibile

Un linguaggio $L \subseteq \Sigma^*$ è **indecidibile** se **non esiste** una TM che lo decide, cioè non esiste una TM M tale che:

- M si ferma sempre su ogni input
- M accetta $w \iff w \in L$

Riduzione mediante Funzione

Un linguaggio A è **riducibile** al linguaggio B ($A \leq_m B$) se esiste una funzione **calcolabile** $f : \Sigma \rightarrow \Sigma$ tale che:

- $\forall w \in \Sigma^*: w \in A \iff f(w) \in B$

Proprietà Fondamentale

Se $A \leq_m B$ e A è indecidibile, allora B è indecidibile.

🔧 METODOLOGIA STANDARD

STEP 1: Identificazione del Problema Sorgente

• **Scegli** un linguaggio A **noto indecidibile** • **Problemi base** da usare come sorgente:

- **ATM** = {⟨M,w⟩ | M è una TM che accetta w}
- **HALTTM** = {⟨M,w⟩ | M è una TM che si ferma su w}
- **ETM** = {⟨M⟩ | M è una TM con L(M) = ∅}
- **REGULARTM** = {⟨M⟩ | M è una TM con L(M) regolare}

STEP 2: Costruzione della Riduzione

• **Definisci** funzione f : istanze di A → istanze di B • **Template**:

```
F = "Su input ⟨istanza_A⟩:
1. [PARSING] Estrai componenti dall'istanza di A
2. [COSTRUZIONE] Costruisci nuova TM M' che:
   • Incorpora il comportamento dell'istanza A
   • Manipola il proprio comportamento per forzare l'appartenenza a B
3. [OUTPUT] Restituisci ⟨M'⟩ o ⟨M', parametri⟩"
```

STEP 3: Dimostrazione di Correttezza

• **Calcolabilità**: f è calcolabile (dare algoritmo esplicito) • **Correttezza bidirezionale**:

- **(⇒)**: istanza_A ∈ A ⟹ f(istanza_A) ∈ B
- **(⇐)**: f(istanza_A) ∈ B ⟹ istanza_A ∈ A

STEP 4: Conclusione

• Poiché A $\leq_m$ B e A indecidibile, allora B è indecidibile

---

📋 RIDUZIONI STANDARD DA ATM

🔸 ATM $\leq_m$ HALTTM

```
PROBLEMA: Dimostrare che HALTTM è indecidibile

RIDUZIONE:
F = "Su input ⟨M,w⟩:
1. Costruisci la seguente TM M':
   M' = "Su input x:
   2. Esegui M su input w
   3. Se M accetta w, ACCETTA
   4. Se M rifiuta w, vai in LOOP infinito"
5. Restituisci ⟨M',w⟩"

CORRETTEZZA:
• Se ⟨M,w⟩ ∈ ATM (M accetta w):
  - M' esegue M su w, M accetta, M' accetta
  - M' si ferma su w, quindi ⟨M',w⟩ ∈ HALTTM
```

```
• Se ⟨M,w⟩ ∉ ATM (M non accetta w):
  - Caso 1: M rifiuta w → M' va in loop → M' non si ferma
  - Caso 2: M loop su w → M' non termina step 1 → M' non si ferma
  - In entrambi i casi ⟨M',w⟩ ∉ HALTTM

CALCOLABILITÀ: F costruisce M' in tempo finito
CONCLUSIONE: HALTTM è indecidibile □
```

◆ ATM $\leq_m$ ETM

```
PROBLEMA: Dimostrare che ETM è indecidibile

RIDUZIONE:
F = "Su input ⟨M,w⟩:
1. Costruisci la seguente TM M':
   M' = "Su input x:
   2. Se x ≠ w, RIFIUTA
   3. Se x = w, esegui M su input w
   4. Se M accetta w, ACCETTA
   5. Se M rifiuta w, RIFIUTA"
6. Restituisci ⟨M'⟩"

CORRETTEZZA:
• Se ⟨M,w⟩ ∈ ATM (M accetta w):
  - M' accetta solo w, quindi L(M') = {w} ≠ ∅
  - Quindi ⟨M'⟩ ∉ ETM

• Se ⟨M,w⟩ ∉ ATM (M non accetta w):
  - M' non accetta nessuna stringa, quindi L(M') = ∅
  - Quindi ⟨M'⟩ ∈ ETM

CONCLUSIONE: ETM è indecidibile □
```

◆ ATM $\leq_m$ EQTM

```
PROBLEMA: Dimostrare che EQTM = {⟨M₁,M₂⟩ | L(M₁) = L(M₂)} è indecidibile

RIDUZIONE:
F = "Su input ⟨M,w⟩:
1. Costruisci TM M₁ che rifiuta tutto: L(M₁) = ∅
2. Costruisci TM M₂:
```

```
   M₂ = "Su input x:
   3. Esegui M su input w
   4. Se M accetta w, ACCETTA x
   5. Se M rifiuta w, RIFIUTA x"
6. Restituisci ⟨M₁,M₂⟩"

CORRETTEZZA:
• Se ⟨M,w⟩ ∈ ATM: L(M₂) = Σ* ≠ ∅ = L(M₁), quindi ⟨M₁,M₂⟩ ∉ EQTM
• Se ⟨M,w⟩ ∉ ATM: L(M₂) = ∅ = L(M₁), quindi ⟨M₁,M₂⟩ ∈ EQTM

CONCLUSIONE: EQTM è indecidibile □
```

🧩 RIDUZIONI AVANZATE E TECNICHE SPECIALI

🔹 Riduzione con Parametri Multipli

Esempio: INFINITETM

```
INFINITETM = {⟨M⟩ | L(M) è infinito}

RIDUZIONE DA ATM:
F = "Su input ⟨M,w⟩:
1. Costruisci TM M':
   M' = "Su input x:
   2. Se x non è della forma 0ⁿ1ⁿ per n ≥ 0, RIFIUTA
   3. Altrimenti, esegui M su input w
   4. Se M accetta w, ACCETTA x
   5. Se M rifiuta w, RIFIUTA x"
6. Restituisci ⟨M'⟩"

CORRETTEZZA:
• Se M accetta w: L(M') = {0ⁿ1ⁿ | n ≥ 0} (infinito)
• Se M non accetta w: L(M') = ∅ (finito)
```

🔹 Costruzioni con Codifica Specifica

Esempio: Linguaggio con Stringa Specifica

```
L₁₀₁₀ = {⟨M⟩ | 1010 ∈ L(M)}

RIDUZIONE DA ATM:
F = "Su input ⟨M,w⟩:
1. Costruisci TM M':
   M' = "Su input x:
   2. Se x ≠ 1010, RIFIUTA
   3. Se x = 1010, esegui M su input w
   4. Se M accetta w, ACCETTA
   5. Se M rifiuta w, RIFIUTA"
6. Restituisci ⟨M'⟩"
```

🔹 Riduzioni con Proprietà Strutturali

Esempio: REGULARTM

```
REGULARTM = {⟨M⟩ | L(M) è regolare}

RIDUZIONE DA ATM:
F = "Su input ⟨M,w⟩:
1. Costruisci TM M':
   M' = "Su input x:
   2. Se x ∈ 0*1*, ACCETTA
   3. Se x = 10ⁿ per qualche n ≥ 0:
      • Esegui M su input w
      • Se M accetta w, ACCETTA
      • Se M rifiuta w, RIFIUTA
   4. Per tutti gli altri x, RIFIUTA"
5. Restituisci ⟨M'⟩"

CORRETTEZZA:
• Se M accetta w: L(M') = 0*1* ∪ {10ⁿ | n ≥ 0} = 0*1* (regolare)
• Se M non accetta w: L(M') = 0*1* (regolare)

PROBLEMA: Entrambi i casi danno linguaggio regolare!

CORREZIONE - Versione corretta:
M' = "Su input x:
1. Se x ∈ 0*1*, ACCETTA
2. Se x = 0ⁿ1ⁿ2ⁿ per qualche n ≥ 1:
   • Esegui M su input w
   • Se M accetta w, ACCETTA
```

```
3. Altrimenti, RIFIUTA"

CORRETTEZZA CORRETTA:
• Se M accetta w: L(M') = 0*1* ∪ {0ⁿ1ⁿ2ⁿ | n ≥ 1} (non regolare)
• Se M non accetta w: L(M') = 0*1* (regolare)
```

⚠️ ERRORI COMUNI DA EVITARE

❌ Riduzione nella Direzione Sbagliata

• **Errore**: Ridurre B $\leq_m$ A invece di A $\leq_m$ B • **Correzione**: Per dimostrare B indecidibile, serve A $\leq_m$ B con A indecidibile

❌ Correttezza Unidirezionale

• **Errore**: Dimostrare solo una direzione dell'equivalenza • **Correzione**: Sempre dimostrare entrambe (⇒) e (⇐)

❌ Funzione Non Calcolabile

• **Errore**: Definire f che non è calcolabile • **Correzione**: Dare algoritmo esplicito per calcolare f

❌ Costruzioni Inconsistenti

• **Errore**: TM costruita non si comporta come previsto • **Correzione**: Verificare attentamente il comportamento in tutti i casi

❌ Parametri Mal Gestiti

• **Errore**: Confondere parametri nella costruzione • **Correzione**: Tenere traccia esplicita di tutti i parametri

---

📝 TEMPLATE GENERICO PER RIDUZIONI

```
PROBLEMA: Dimostrare che L è indecidibile

TEOREMA: L è indecidibile.

DIMOSTRAZIONE: Mostriamo ATM ≤ₘ L.
```

```
RIDUZIONE:
Definiamo la seguente funzione calcolabile f:

F = "Su input ⟨M,w⟩:
1. [COSTRUZIONE MACCHINA AUSILIARIA]
   Costruisci la seguente TM M':

   M' = "Su input x:
   2. [CONDIZIONI SU INPUT]
      • Se x soddisfa condizione_1, [comportamento_1]
      • Se x soddisfa condizione_2, [comportamento_2]
      • ...
   3. [SIMULAZIONE ORIGINALE]
      • Esegui M su input w
      • Se M accetta w, [azione_accettazione]
      • Se M rifiuta w, [azione_rifiuto]"

4. [OUTPUT]
   Restituisci ⟨M', parametri_aggiuntivi⟩"

CORRETTEZZA:
• CALCOLABILITÀ: F è chiaramente calcolabile in tempo finito

• DIREZIONE (⇒): Se ⟨M,w⟩ ∈ ATM, allora f(⟨M,w⟩) ∈ L
  Dimostrazione: [argomento specifico]

• DIREZIONE (⇐): Se f(⟨M,w⟩) ∈ L, allora ⟨M,w⟩ ∈ ATM
  Dimostrazione: [argomento specifico]

CONCLUSIONE:
Poiché ATM ≤ₘ L e ATM è indecidibile, allora L è indecidibile. □
```

🎯 PROBLEMI DI RIFERIMENTO INDECIDIBILI

Base (sempre indecidibili):

- **ATM** = {⟨M,w⟩ | M accetta w}
- **HALTTM** = {⟨M,w⟩ | M si ferma su w}
- **ETM** = {⟨M⟩ | L(M) = ∅}

Derivati (dimostrati per riduzione):

- **EQTM** = $\{\langle M_1, M_2\rangle \mid L(M_1) = L(M_2)\}$
- **ALLTM** = $\{\langle M\rangle \mid L(M) = \Sigma^*\}$
- **REGULARTM** = $\{\langle M\rangle \mid L(M)$ è regolare$\}$
- **CFTM** = $\{\langle M\rangle \mid L(M)$ è context-free$\}$
- **FINITETM** = $\{\langle M\rangle \mid L(M)$ è finito$\}$

📊 CHECKLIST FINALE

Prima di consegnare, verifica:

- [ ] Linguaggio L definito formalmente
- [ ] Problema sorgente A identificato (e noto indecidibile)
- [ ] Funzione f definita esplicitamente
- [ ] Algoritmo per calcolare f dato completamente
- [ ] Entrambe le direzioni della correttezza dimostrate
- [ ] Calcolabilità di f argomentata
- [ ] Costruzione TM ausiliaria corretta e completa
- [ ] Comportamento TM verificato in tutti i casi
- [ ] Conclusione esplicita di indecidibilità

Strategie di Debugging:

- [ ] Testare riduzione su esempi piccoli
- [ ] Verificare comportamento TM in casi limite
- [ ] Controllare che parametri siano gestiti correttamente
- [ ] Assicurarsi che direzione riduzione sia corretta