

Revisão: Redutibilidade, Completude e Indecidibilidade

Computabilidade e Complexidade

Márcio Nicolau

2025-10-20

Table of contents

Objetivos da Aula	1
Conteúdo	1
O Mapa da Computabilidade: Onde Estamos?	1
Redutibilidade: A Ferramenta Universal	2
Exemplo Prático: Provando que E_{TM} é Indecidível	2
Completude: O Cume da Montanha	4
Exercícios de Verificação	4
Referências Bibliográficas	4

List of Figures

1	Passo a passo para provar que um novo problema (L) é indecidível.	3
2	A _{TM} é o problema RE-Completo do qual a indecidibilidade se espalha para outros problemas via reduções.	5

Objetivos da Aula

- Consolidar a habilidade de classificar problemas na hierarquia de decidibilidade (R, RE, co-RE, não-RE).
- Praticar a técnica de redução como a principal ferramenta para provar indecidibilidade.
- Reforçar o significado e a importância de problemas completos para uma classe.
- Aplicar o conhecimento teórico para analisar a computabilidade de novos problemas.

Conteúdo

O Mapa da Computabilidade: Onde Estamos?

Após várias aulas, temos um mapa claro do universo dos problemas computacionais. A habilidade mais crucial nesta etapa do curso é saber localizar um novo problema neste mapa.

Definições Essenciais (Revisão)

- **Decidível (R)**: Existe um algoritmo que **sempre para** com a resposta correta (sim/não). É a nossa “terra firme”.
- **Reconhecível (RE)**: Existe um algoritmo que garante parar com uma resposta “sim” para instâncias positivas, mas pode entrar em loop para instâncias negativas.
- **Co-Reconhecível (co-RE)**: O complemento do problema é RE. O algoritmo garante parar com uma resposta “não”, mas pode entrar em loop para instâncias positivas.
- **Indecidível**: Qualquer problema que não está em R.
- **Não-Reconhecível**: Problemas que não são nem RE nem co-RE. A “terra incógnita”.
- **Teorema Fundamental**: $R = RE \cap co-RE$. Se um problema e seu complemento são ambos reconhecíveis, então o problema é decidível.

Redutibilidade: A Ferramenta Universal

A redutibilidade ($A \leq_m B$) é a nossa ferramenta mais poderosa. Ela nos permite transferir a “dificuldade” de um problema conhecido para um novo.

A Lógica Central: Se A é difícil e eu posso usar uma solução para B para resolver A, então B deve ser pelo menos tão difícil quanto A.

Receita para Provar que uma Linguagem L é Indecidível

1. **Escolha um Problema Semente**: Selecione uma linguagem S que você **já sabe** que é indecidível. Quase sempre, a melhor escolha é A_{TM} .
2. **Construa a Redução**: Mostre que $S \leq_m L$. Para isso, você precisa construir uma **função computável** f que transforma uma entrada w_S de S em uma entrada w_L de L .
3. **A Construção de f** : Geralmente, f recebe uma instância de S (ex: $\langle M, w \rangle$) e produz a descrição de uma **nova Máquina de Turing** M' como saída. O comportamento de M' é projetado para depender do resultado da computação de M em w .
4. **Prove a Equivalência**: Demonstre que a construção funciona nos dois sentidos: $w_S \in S \iff f(w_S) \in L$.
5. **Conclua**: Como S é indecidível e S se reduz a L , então L **deve ser indecidível**.

Diagrama: O Fluxo de uma Prova por Redução

Exemplo Prático: Provando que E_{TM} é Indecidível

Vamos usar a receita para provar que $E_{TM} = \{\langle M \rangle \mid L(M) = \emptyset\}$ é indecidível.

1. **Problema Semente**: A_{TM} .
2. **Objetivo**: Mostrar que $A_{TM} \leq_m E_{TM}$. Ops, isso parece difícil. A pergunta em A_{TM} é sobre aceitar *uma* string, enquanto E_{TM} é sobre aceitar *nenhuma*. Elas parecem opostas. Vamos tentar reduzir ao complemento, $\overline{E_{TM}} = \{\langle M \rangle \mid L(M) \neq \emptyset\}$.

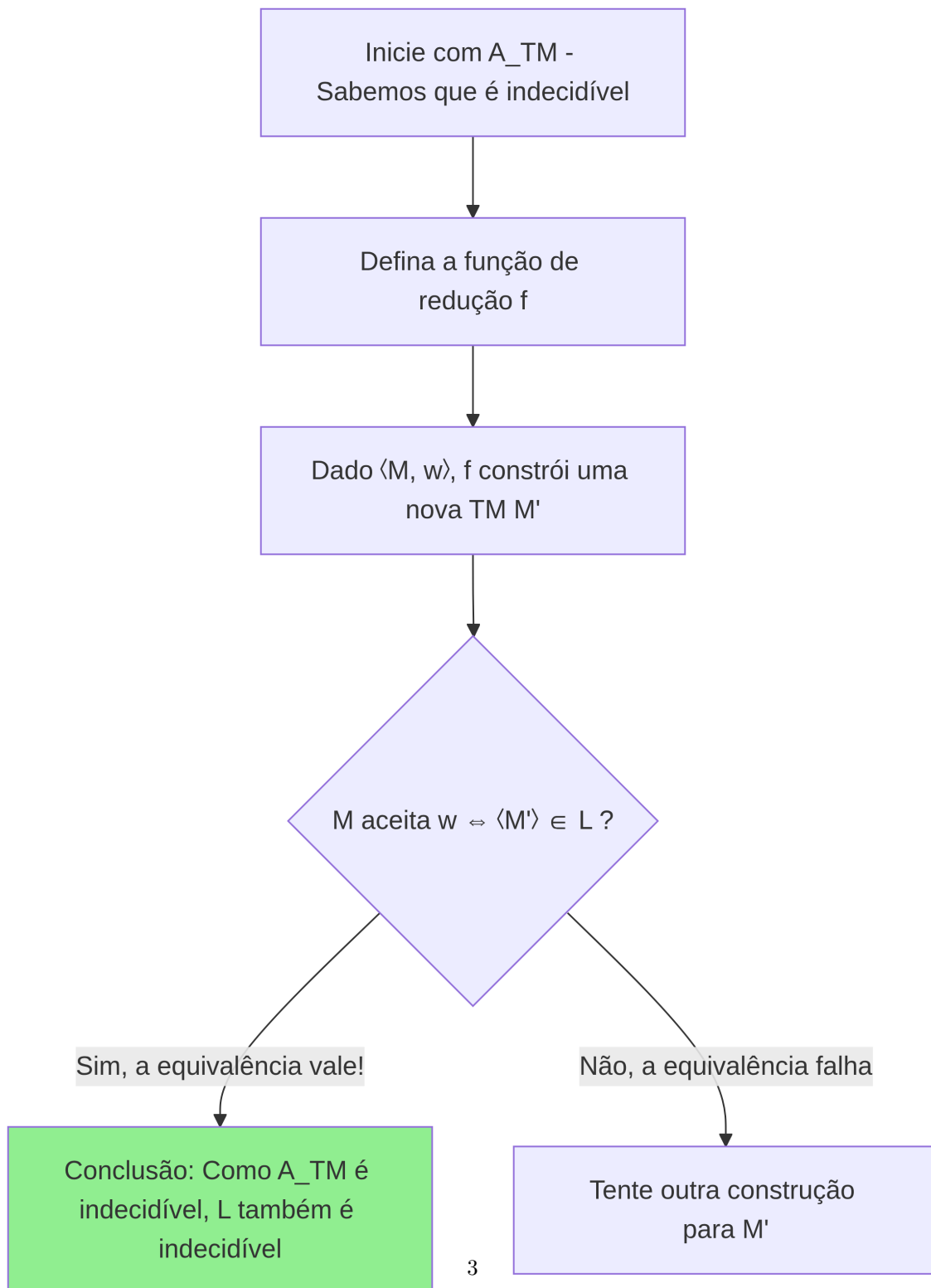


Figure 1: Passo a passo para provar que um novo problema (L) é indecidível.

3. **Construção da Redução (f):** A função f recebe uma entrada $\langle M, w \rangle$ para A_{TM} e produz uma saída $\langle M' \rangle$ para $\overline{E_{TM}}$.

M' (recebe uma entrada x):

1. Ignore completamente a entrada x .
2. Simule a execução de M na entrada w (a original).
3. Se a simulação de M aceita w , então M' aceita x .
4. Se a simulação de M rejeita ou entra em loop, M' também rejeita ou entra em loop.

4. **Prova da Equivalência:**

- ($\Rightarrow$): Se $\langle M, w \rangle \in A_{TM}$ (M aceita w), então M' em sua etapa 3 aceitará *qualquer* entrada x . Portanto, $L(M') = \Sigma^*$, que não é vazia. Logo, $\langle M' \rangle \in \overline{E_{TM}}$.
- ($\Leftarrow$): Se $\langle M, w \rangle \notin A_{TM}$ (M não aceita w), então a simulação em M' nunca chegará a um estado de aceitação. Portanto, M' nunca aceita sua entrada x . Logo, $L(M') = \emptyset$. Logo, $\langle M' \rangle \notin \overline{E_{TM}}$.

5. **Conclusão:** Provamos que $A_{TM} \leq_m \overline{E_{TM}}$. Como A_{TM} é indecidível, $\overline{E_{TM}}$ também é indecidível. Isso também nos diz que, como $\overline{E_{TM}}$ é reconhecível, ela deve estar em **RE**. Pelo Teorema Fundamental, isso implica que seu complemento, E_{TM} , **não pode ser reconhecível (não está em RE)**.

Completeness: O Cume da Montanha

Um problema ser **completo** para uma classe significa que ele captura a essência da dificuldade de toda a classe.

- **RE-Completo:** Os problemas mais difíceis que ainda são semi-decidíveis (reconhecíveis). A_{TM} e $HALT_{TM}$ são os exemplos canônicos. Se você pudesse resolver A_{TM} , poderia resolver *qualquer* problema reconhecível.
- **co-RE-Completo:** Os problemas cujos complementos são RE-completos. Ex: E_{TM} .

Diagrama: Mapa de Reduções e Completeness

Exercícios de Verificação

Atividade Prática de Revisão

1. **Classificação Rápida:** Classifique as seguintes linguagens e justifique brevemente:
 - $L_1 = \{\langle G \rangle \mid G \text{ é uma Gramática Livre de Contexto e } L(G) = \emptyset\}$.
 - $L_2 = \{\langle M \rangle \mid M \text{ é uma TM que tem exatamente 10 estados}\}$.
 - $L_3 = \{\langle M \rangle \mid M \text{ é uma TM que aceita a string 'aba'}\}$.
2. **Construção de Redução:** Prove que a linguagem $REGULAR_{TM} = \{\langle M \rangle \mid L(M) \text{ é uma linguagem regular}\}$ é indecidível. Siga a “receita” e construa uma redução a partir de A_{TM} .
3. **Completeness e Decidibilidade:** Se uma linguagem L é RE-Completa, ela pode ser decidível? E se ela for co-RE-Completa? E se ela for tanto RE-Completa quanto co-RE-Completa? Explique.

Referências Bibliográficas

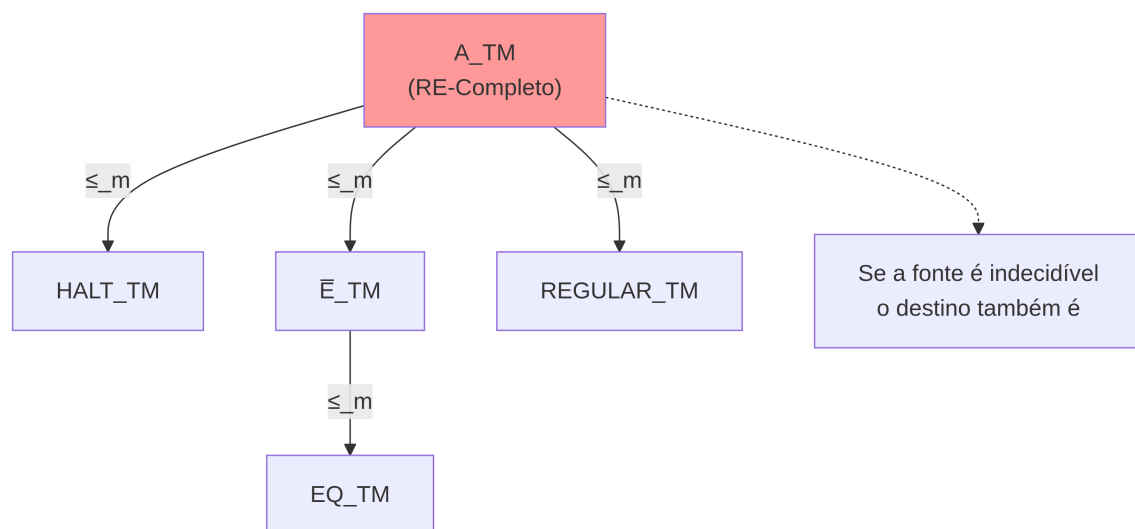


Figure 2: A_{TM} é o problema RE-Completo do qual a indecidibilidade se espalha para outros problemas via reduções.