Introdução à redutibilidade

→ Redução é uma técnica de projeto de ~~algoritmos~~ máquinas de Turing em que se usa uma máquina feita para um ~~problema~~ linguagem B para criar uma máquina para A.

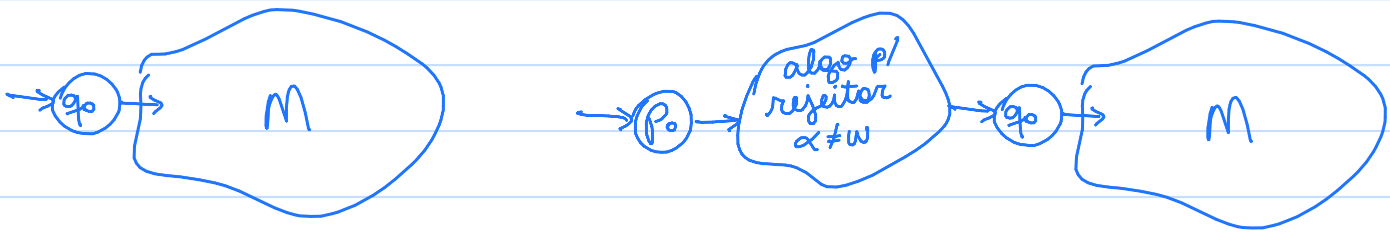
M_A = "Sobre a entrada x:
(1) Seja $y = f(x)$
(2) Rode M_B sobre y
(3) Aceite se M_B aceita. Rejeite se M_B rejeita"

→ Reduzir de A para B:

$x \xrightarrow{f} y \longrightarrow \boxed{M_B} \longrightarrow$ aceita / rejeita

DEFINIÇÃO: Uma função $f: \Sigma^* \longrightarrow \Sigma^*$ é computável se alguma máquina de Turing a computa: w começa na fita, a máquina executa e para, contendo $f(w)$ na fita.

Exemplos: $f(w) = \$w$, com w qualquer
$f(\langle x, y\rangle) = \langle x+y \rangle$, com x e y inteiros
$f(\langle M, w\rangle) = \langle M', w\rangle$, com M uma MT e w uma cadeia e e M' é M acrescida de estados iniciais para rejeitar toda $\alpha \neq w$.

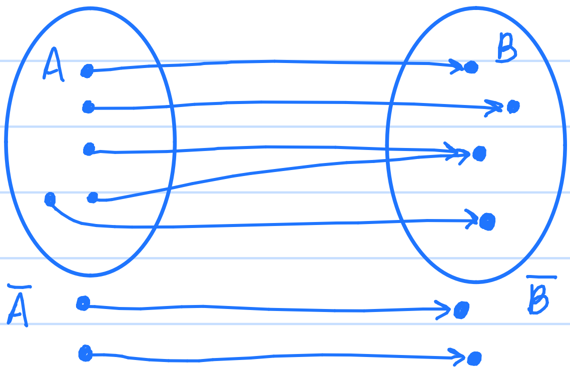


$((q_0, \dots, q_A, q_R), (x_1, \dots, x_m), (y_1, \dots, y_m, \sqcup), ((q_0, x_1, q_1), \dots, (q_R, \sqcup, q_R)), q_0, q_A, q_R)\ (x_3\, x_4\, x_6\, x_1)$

$((p_0, \dots, q_0, \dots, q_A, q_R), (x_1, \dots, x_m), (y_1, \dots, y_t, \sqcup), ((p_0, x_1, q_R), \dots, (q_0, x_1, q_1), \dots, (q_R, \sqcup, q_R)), p_0, q_A, q_R)\ (x_3\, x_4\, x_6\, x_1)$

DEFINIÇÃO: A linguagem A é *redutível* à linguagem B se existe uma função computável $f: \Sigma^* \to \Sigma^*$ tal que para toda $w \in \Sigma^*$

$$w \in A \text{ se e somente se } f(w) \in B.$$



$w \in A \Leftrightarrow f(w) \in B$

$w \notin A \Leftrightarrow f(w) \notin B$ → $\bar{A}$ reduz para $\bar{B}$

ATENÇÃO!!!

→ Ao criar uma redução f de A para B, precisamos mostrar que "$w \in A \Leftrightarrow f(w) \in B$" é verdadeira.

Para isso, sempre mostramos (1) e (2):

(1) $w \in A \Rightarrow f(w) \in B$

(2) $f(w) \in B \Rightarrow w \in A$

Como (2) é equivalente a

(2)' $w \notin A \Rightarrow f(w) \notin B$

o que mostramos, na verdade, é (1) e (2)'.

→ É muito mais intuitivo considerar os caso "$w \in A$" e "$w \notin A$".

Exemplos de redução

Lembre:

$$\begin{cases} A_{AFD} = \{\langle A, w\rangle : A \text{ é AFD que aceita } w\} \\ A_{AFN} = \{\langle A, w\rangle : A \text{ é AFN que aceita } w\} \\ A_{ER} = \{\langle R, w\rangle : R \text{ é ER que descreve } w\} \end{cases}$$

Criamos $M_{A_{AFD}}$, decisora para A_{AFD}, diretamente.
Reduzimos A_{AFN} para A_{AFD} : em $M_{A_{AFN}}$, a conversão do AFN no AFD é a redução:

f = "Sobre a entrada $\langle A, w \rangle$, onde A é AFN e w é cadeia:
(1) Converta A em um AFD equivalente B.
(2) Devolva $\langle B, w \rangle$."

Para mostrar que f é redução, devemos mostrar que
$\langle A,w \rangle \in A_{AFN}$ sse $f(\langle A,w \rangle) = \langle B,w \rangle \in A_{AFD}$

1. Vamos mostrar que se $\langle A,w \rangle \in A_{AFN}$, então $\langle B,w \rangle \in A_{AFD}$.
Se $\langle A,w \rangle \in A_{AFN}$, então o AFN A aceita w.
Como B é um AFD equivalente a A, então B aceita w.
Logo, $\langle B,w \rangle \in A_{AFD}$.

2. Vamos mostrar que se $\langle A,w \rangle \notin A_{AFN}$, então $\langle B,w \rangle \notin A_{AFD}$.
Se $\langle A,w \rangle \notin A_{AFN}$, então o AFN A não aceita w.
Como B é um AFD equivalente a A, então B não aceita w.
Logo, $\langle B,w \rangle \notin A_{AFD}$.

Agora que sabemos que F é redução, fica mais claro que M_{AFN} decide a linguagem A_{AFN}:

$M_{A_{AFN}}$ = "Sobre a entrada $\langle A, w \rangle$, onde A é AFN e w é cadeia:
(1) Seja $\langle B,w \rangle = f(\langle A,w \rangle)$.
(2) Execute $M_{A_{AFD}}$ sobre $\langle B,w \rangle$.
(3) Se $M_{A_{AFD}}$ aceitar, aceite. Caso contrário, rejeite."

Também já reduzimos A_{ER} para A_{AFN}.

f = "Sobre a entrada $\langle R, w \rangle$, onde R é ER e w é cadeia:
(1) Converta R em um AFN equivalente B.
(2) Devolva $\langle B, w \rangle$."

Para mostrar que f é redução, devemos mostrar que

$\langle R,w\rangle \in A_{ER}$ sse $f(\langle R,w\rangle) = \langle B,w\rangle \in A_{AFN}$

① Vamos mostrar que se $\langle R,w\rangle \in A_{ER}$, então $\langle B,w\rangle \in A_{AFN}$

Se $\langle R,w\rangle \in A_{ER}$, então a ER R descreve w.

Como B é um AFN equivalente a R, B aceita w.

Logo, $\langle B,w\rangle \in A_{AFN}$.

② Vamos mostrar que se $\langle R,w\rangle \notin A_{ER}$, então $\langle B,w\rangle \notin A_{AFN}$

Se $\langle R,w\rangle \notin A_{ER}$, então a ER R não descreve w.

Como B é um AFN equivalente, então B não aceita w.

Logo, $\langle B,w\rangle \notin A_{AFN}$.

Com isso, $M_{A_{ER}}$ decide A_{ER}:

$M_{A_{ER}}$ = "Sobre a entrada $\langle R,w\rangle$, onde R é ER e w é cadeia:

(1) Seja $\langle B,w\rangle = f(\langle R,w\rangle)$.

(2) Execute $M_{A_{AFN}}$ sobre $\langle B,w\rangle$.

(3) Se $M_{A_{AFN}}$ aceitar, aceite. Caso contrário, rejeite."

Observe que, indiretamente, reduzimos A_{ER} para A_{AFD} (composição).

Lembre: $\begin{cases} V_{AFD} = \{\langle A\rangle : A \text{ é AFD e } L(A)=\emptyset\} \\ E_{AFD} = \{\langle A,B\rangle : A \text{ e } B \text{ são AFDs e } L(A)=L(B)\} \end{cases}$

Já reduzimos E_{AFD} para V_{AFD}:

f = "Sobre a entrada $\langle A,B\rangle$, onde A e B são AFDs:

(1) Construa um AFD C tal que $L(C) = \left(L(A)\cap\overline{L(B)}\right)\cup\left(\overline{L(A)}\cap L(B)\right)$

(2) Devolva $\langle C\rangle$."

Para mostrar que f é redução, devemos mostrar que

$\langle A,B\rangle \in E_{AFD}$ sse $f(\langle A,B\rangle) = \langle C\rangle \in V_{AFD}$

① Vamos mostrar que se $\langle A,B\rangle \in E_{AFD}$, então $\langle C\rangle \in V_{AFD}$.

Se $\langle A,B\rangle \in E_{AFD}$, então $L(A) = L(B)$.

Mas então $\overline{L(A)} = \overline{L(B)}$, $L(A)\cap\overline{L(B)} = \emptyset$ e $\overline{L(A)}\cap L(B) = \emptyset$.

Logo, $L(C) = \emptyset$ e, portanto, $\langle C\rangle \in V_{AFD}$.

(2) Vamos mostrar que se $\langle A,B \rangle \notin E_{AFD}$, então $\langle C \rangle \notin V_{AFD}$.
Se $\langle A,B \rangle \notin E_{AFD}$, então $L(A) \neq L(B)$ e, logo, $\overline{L(A)} \neq \overline{L(B)}$.
Mas então $L(A) \cap \overline{L(B)} \neq \emptyset$ ou $\overline{L(A)} \cap L(B) \neq \emptyset$, pois se tivéssemos $L(A) \cap \overline{L(B)} = \emptyset$ e $\overline{L(A)} \cap L(B) = \emptyset$, teríamos $L(A) \subseteq L(B)$ e $L(B) \subseteq L(A)$, o que implicaria $L(A) = L(B)$.
Com $L(A) \cap \overline{L(B)} \neq \emptyset$ ou $\overline{L(A)} \cap L(B) \neq \emptyset$, $L(C) \neq \emptyset$, ou seja, $\langle C \rangle \notin V_{AFD}$.

Com isso, temos que $M_{E_{AFD}}$ decide E_{AFD}:
$M_{E_{AFD}}$ = "Sobre a entrada $\langle A,B \rangle$:
(1) Seja $\langle C \rangle = f(\langle A,B \rangle)$.
(2) Execute $M_{V_{AFD}}$ sobre $\langle C \rangle$.
(3) Se $M_{V_{AFD}}$ aceita, aceite. Caso contrário, rejeite."

Usando redução para provar indecidibilidade

$P_{MT} = \{ \langle M,w \rangle : M$ é MT que para sobre $w \}$ é decidível?
Suponha que seja. Então existe MT R que decide P_{MT}.
Construa a seguinte máquina para A_{MT}:
S = "Sobre a entrada $\langle M,w \rangle$, onde M é MT e w é cadeia:
(1) Execute R sobre $\langle M,w \rangle$.
(2) Se R rejeita, rejeite.
(3) Se R aceita, simule M sobre w. Se M aceita, aceite. Caso contrário, rejeite."

Como R "existe", $R(\langle M,w \rangle)$ = rejeita implica que M não para sobre w e portanto $\langle M,w \rangle \notin A_{MT}$.
Se $R(\langle M,w \rangle)$ = aceita, então M para sobre w.
Nesse último caso, simulamos M sobre w e aceitamos se M aceita w ou rejeitamos caso contrário.
Mas então a máquina S é uma decisora para A_{MT}, o que é uma contradição.

O que ganhamos com a redução?

TEOREMA: Se A reduz para B e

1) B é decidível, então A é decidível.
2) B é Turing-reconhecível, então A é Turing-reconhecível.

DEMONSTRAÇÃO: Seja f a redução de A para B. Então:

M_A = "Sobre a entrada x:

(1) Seja $y = f(x)$
(2) Rode M_B sobre y
(3) Dê como saída o que M_B der como saída" CQD

COROLÁRIO: Se A reduz para B e

1) A é indecidível, então B é indecidível.
2) A é Turing-irreconhecível, então B é Turing-irreconhecível.

Usando redução para provar indecidibilidade

$V_{MT} = \{ \langle M \rangle : M$ é MT e $L(M) = \phi \}$ é decidível?

Como usar V_{MT} para resolver A_{MT}?

Como criar f tal que $\langle M, w \rangle \in A_{MT}$ sse $f(\langle M,w \rangle) = \langle M' \rangle \in V_{MT}$?

Dados M e w, quem deve ser M' para que

$L(M') = \phi \Rightarrow M$ aceita w

$L(M') \neq \phi \Rightarrow M$ não aceita w ?

Seja F = "Sobre a entrada $\langle M, w \rangle$, onde M é MT e w é cadeia:

(1) Construa M' = "Sobre a entrada x:

(1) Se $x \neq w$, rejeite.

(2) Se $x = w$, rode M sobre w e aceite se M aceitar."

(2) Devolva $\langle M' \rangle$"

F é redução de A_{MT} para $\overline{V_{MT}}$, pois, seja $\langle M' \rangle = F\langle M, w \rangle$:

1) Se $\langle M, w \rangle \in A_{MT}$, então M aceita w. Pela construção, $L(M') = \{w\}$. Então $\langle M' \rangle \notin V_{MT}$.

2) Se $\langle M, w \rangle \notin A_{MT}$, então M não aceita w. Pela construção, M' rejeita tudo. Então $L(M') = \phi$ e $\therefore \langle M' \rangle \in V_{MT}$.

Suponha que R seja decisora para V_{MT}. Construa S para A_{MT}:

S = "Sobre a entrada $\langle M, w \rangle$, onde M é MT e w é cadeia:

(1) Seja $\langle M' \rangle = F(\langle M, w \rangle)$.

(2) Rode R sobre $\langle M' \rangle$.

(3) Se R aceita, rejeite. Se R rejeita, aceite."

Como R decide V_{MT}, ela sempre aceita ou rejeita. Claramente, S também sempre aceita ou rejeita e é, portanto, decisora. Mas então A_{MT} é decidível, absurdo.