Linguagens Formais e Autômatos

P. Blauth Menezes

blauth@inf.ufrgs.br

Departamento de Informática Teórica

Instituto de Informática / UFRGS

Linguagens Formais e Autômatos

P. Blauth Menezes

8 - Linguagens Recursivamente Enumeráveis e Sensíveis ao Contexto

8 - Linguagens Recursivamente Enumeráveis e Sensíveis ao Contexto

8 Linguagens Recursivamente Enumeráveis e Sensíveis ao Contexto

◆ **Ciência da Computação**

- conhecimento sistematizado relativo à computação

◆ **Origem da Ciência da Computação: remota**

- antiga Grécia: III a.C - desenho de algoritmos por Euclides
- Babilônia: estudos sobre complexidade e reducibilidade de problemas
- início do século XX: pesquisas com o objetivo de definir
  - ∗ modelo computacional suficientemente genérico
  - ∗ capaz de implementar qualquer função computável

◆ Alan Turing (1936) propôs um modelo

- Máquina de Turing

- aceito como uma formalização de
  - procedimento efetivo
  - algoritmo ou
  - função computável

- algoritmo
  - seqüência finita de instruções
  - podem ser realizadas mecanicamente
  - em um tempo finito

◆ Alonzo Church (1936)

- Hipótese de Church

qualquer função computável pode ser processada por uma máquina de Turing

  - ∗ existe um procedimento expresso na forma de uma máquina de Turing capaz de processar a função

- como a noção intuitiva de procedimentos *não* é matematicamente precisa
  - ∗ impossível demonstrar formalmente se a máquina de Turing é, de fato, o mais genérico dispositivo de computação
  - ∗ *mostrado*: todos os demais modelos propostos possuem, no máximo, a mesma capacidade computacional

◆ Resumidamente, uma máquina de Turing

- autômato
- fita não possui tamanho máximo
- pode ser usada simultaneamente como dispositivo de entrada, de saída e de memória de trabalho

◆ Linguagens Recursivamente Enumeráveis ou Tipo 0

- aceitas por uma máquina de Turing
- segundo a Hipótese de Church, a Classe das Linguagens Recursivamente Enumeráveis
  - ∗ conjunto de todas as linguagens
  - ∗ que podem ser reconhecidas mecanicamente
  - ∗ em um tempo finito

◆ Gramática Irrestrita

- sem restrições sobre a forma das produções
- mesmo poder computacional que o formalismo Máquina de Turing

◆ Conseqüência importante do estudo das linguagens recursivamente enumeráveis

existem mais problemas não-solucionáveis do que problemas solucionáveis

◆ Classe das Linguagens Recursivamente Enumeráveis

- inclui algumas para as quais é
  - impossível determinar mecanicamente
  - se uma palavra *não* pertence à linguagem
- se L é uma destas linguagens, então
  - para qualquer máquina de Turing M que aceita L
  - existe pelo menos uma palavra w não pertencente a L que
  - ao ser processada por M, a máquina entra em *loop* infinito
- ou seja
  - se w pertence a L, M pára e aceita a entrada
  - se w *não* pertence a L, M pode parar, rejeitando a palavra *ou* permanecer processando indefinidamente

◆ Linguagens Recursivas

- subclasse da Classe das Linguagens Enumeráveis Recursivamente

- existe pelo menos uma máquina de Turing que pára para qualquer entrada, aceitando ou rejeitando

◆ Linguagens Sensíveis ao Contexto ou Tipo 1

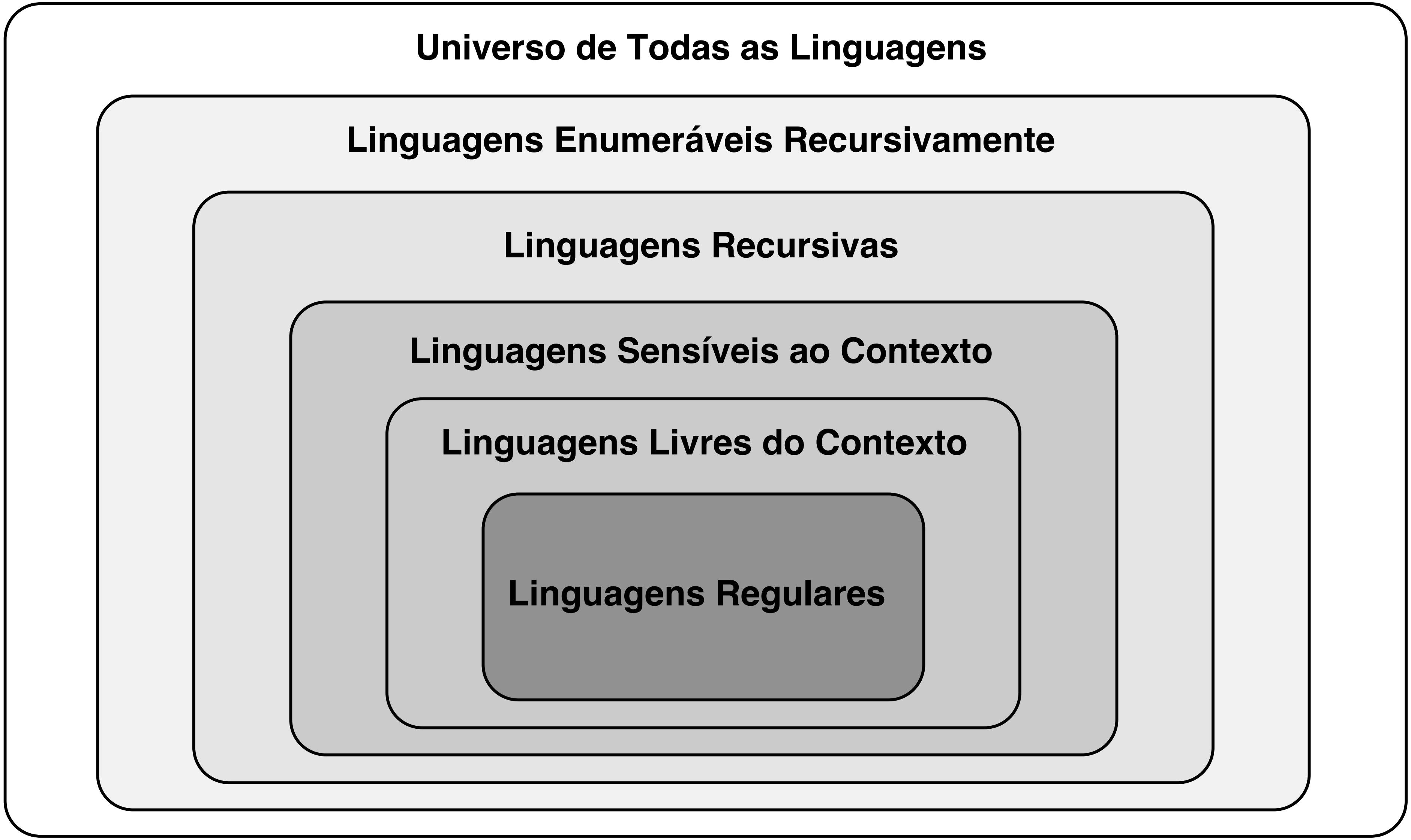
- aceitas por uma Máquina de Turing com Fita Limitada
  - máquina de Turing com limitação no tamanho da fita (finita)
  - exercício: diferença para o Autômato Finito?

◆ Gramática Sensível ao Contexto

- em oposição a “livre do contexto”: lado esquerdo das produções
  - pode ser uma palavra de variáveis ou terminais
  - definindo um "contexto" de derivação

◆ Classe das Linguagens Sensíveis ao Contexto

- contida propriamente na Classe das Linguagens Recursivas
- classe especialmente importante
  - inclui a grande maioria das linguagens aplicadas


Universo de Todas as Linguagens
Linguagens Enumeráveis Recursivamente
Linguagens Recursivas
Linguagens Sensíveis ao Contexto
Linguagens Livres do Contexto
Linguagens Regulares

8 - Linguagens Recursivamente Enumeráveis e Sensíveis ao Contexto

8.1 Máquina de Turing

- **Noção de algoritmo não é matematicamente precisa**
- **Intuitivamente, deve possuir**
  - descrição finita
  - passos
    - discretos (em oposição ao contínuo)
    - executáveis mecanicamente
    - em um tempo finito

8 - Linguagens Recursivamente Enumeráveis e Sensíveis ao Contexto

8.1.1 Noção Intuitiva

- **Máquina de Turing (Alan Turing, 1936)**
  - mecanismo simples
  - formaliza a idéia de uma pessoa que realiza cálculos
  - lembra os computadores atuais
    - embora proposta anos antes do primeiro computador digital
- **Modelo máquina de Turing**
  - no mínimo, mesmo poder computacional
  - de qualquer computador de propósito geral

◆ Ponto de partida de Turing

- uma pessoa
- com um instrumento de escrita e um apagador
- realiza cálculos em uma folha de papel, organizada em quadrados

◆ Inicialmente, a folha de papel

- contém somente os dados iniciais do problema

◆ Trabalho da pessoa: seqüências de operações simples

- ler um símbolo de um quadrado
- alterar um símbolo em um quadrado
- mover os olhos para outro quadrado
- fim dos cálculos
  - representação satisfatória para a resposta desejada

◆ Hipóteses aceitáveis

- natureza bidimensional do papel *não* é essencial para os cálculos
  - fita infinita organizada em quadrados
- conjunto de símbolos: finito
  - possível utilizar seqüências de símbolos

◆ Hipóteses aceitáveis (pessoa)

- conjunto de estados da mente durante o cálculo
  - finito
  - dois em particular: "estado inicial" e "estado final"

- o comportamento, a cada momento, é determinado pelo
  - estado presente
  - símbolo para o qual sua atenção está voltada

- pessoa é capaz de
  - observar e alterar o símbolo de apenas um quadrado
  - transferir sua atenção para um dos quadrados adjacentes

8 - Linguagens Recursivamente Enumeráveis e Sensíveis ao Contexto

8.1.2 Modelo

◆ Constituído de três partes

- Fita, usada simultaneamente como dispositivo de
  - entrada, saída e memória de trabalho
- Unidade de Controle
  - reflete o estado corrente da máquina
  - possui uma unidade de leitura e gravação (cabeça da fita)
  - acessa uma célula da fita de cada vez
  - se movimenta para a esquerda ou para a direita
- Programa, Função Programa ou Função de Transição
  - define: estado da máquina
  - comanda: leituras, gravações e sentido de movimento (cabeça)

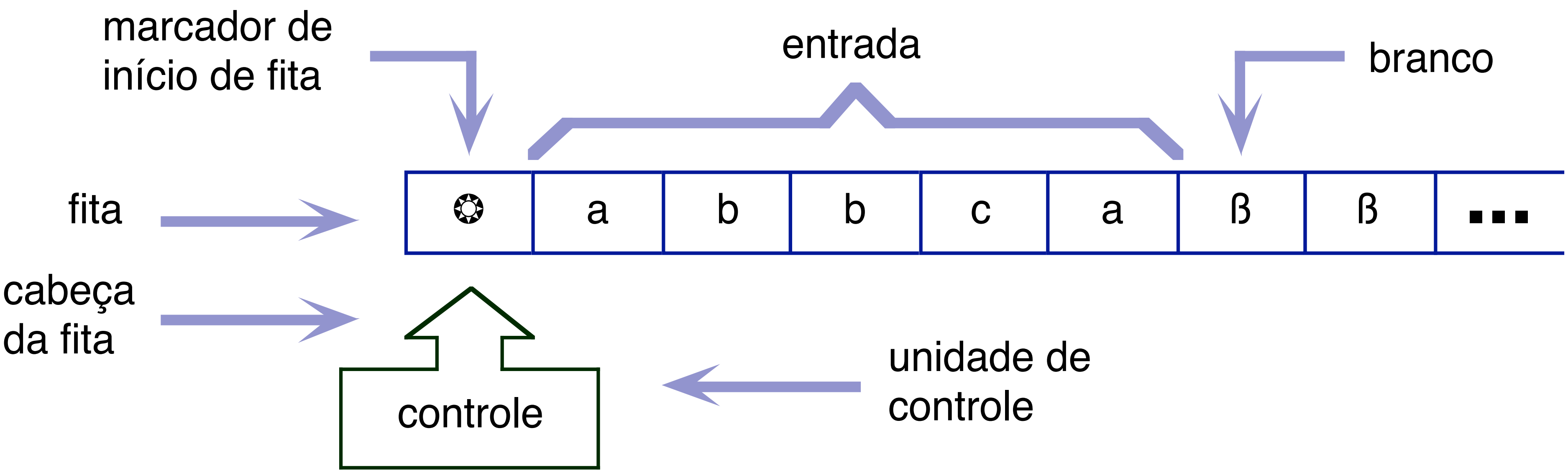
◆ Fita: finita à esquerda e *infinita* à direita

- infinita: "tão grande quanto necessário"
- dividida em células, cada uma armazenando um símbolo

◆ Símbolos podem

- pertencer ao alfabeto de entrada
- pertencer ao alfabeto auxiliar
- ser "branco"
- ser "marcador de início de fita"

- Inicialmente
  - palavra a ser processada: células mais à esquerda (após o marcador de início de fita)
  - demais células: "branco"



◆ Unidade de controle

- número *finito* e *predefinido* de estados
- cabeça da fita
  - lê um símbolo de cada vez e grava um novo símbolo
  - move uma célula para a direita ou para a esquerda
- símbolo gravado e o sentido do movimento
  - definidos pelo programa

Def: Máquina de Turing

$$M = (\Sigma, Q, \delta, q_0, F, V, \beta, ❂)$$

- Σ - alfabeto (de símbolos) de entrada
- Q - conjunto de estados possíveis da máquina (finito)
- δ - (função) programa ou função de transição (função parcial)
  - suponha que $\Sigma \cup V$ e $\{ \beta, ❂ \}$ são conjuntos disjuntos

    $$\delta: Q \times (\Sigma \cup V \cup \{ \beta, ❂ \}) \rightarrow Q \times (\Sigma \cup V \cup \{ \beta, ❂ \}) \times \{ E, D \}$$

  - transição da máquina: $\delta(p, x) = (q, y, m)$
- q_0 - estado inicial: elemento distinguido de Q
- F - conjunto de estados finais: subconjunto de Q
- V - alfabeto auxiliar (pode ser vazio)
- β - símbolo especial branco
- ❂ - símbolo de início ou marcador de início da fita

◆ Símbolo de início de fita

- ocorre exatamente uma vez e na célula mais à esquerda da fita

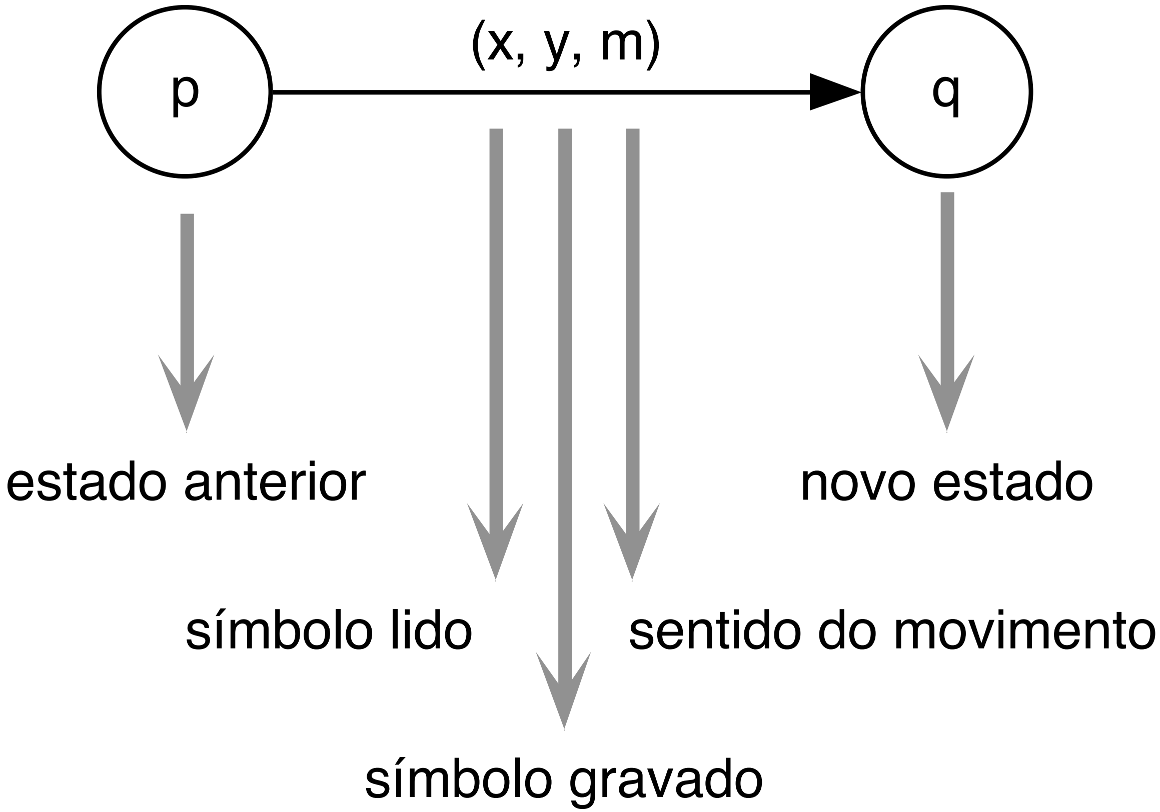
◆ Função programa

- considera
  - ⁎ estado corrente
  - ⁎ símbolo lido da fita

- determina
  - ⁎ novo estado
  - ⁎ símbolo a ser gravado
  - ⁎ sentido de movimento da cabeça (E e D)

◆ Função programa interpretada como um diagrama

- estados inicial e finais: como nos autômatos finitos
- suponha a transição $\delta(p, x) = (q, y, m)$)



◆ Computação de uma máquina de Turing M, para uma palavra de entrada w

- sucessiva aplicação da função programa
  - ∗ a partir do estado inicial
  - ∗ cabeça posicionada na célula mais à esquerda da fita
  - ∗ até ocorrer uma condição de parada

- processamento pode
  - ∗ parar ou
  - ∗ ficar processando indefinidamente (ciclo ou *loop* infinito)

◆ Aceita a entrada w

- atinge um estado final
  - máquina pára
  - w é aceita

◆ Rejeita a entrada w

- função programa é indefinida para o argumento (símbolo lido e estado corrente)
  - máquina pára
  - w é rejeitada
- argumento define um movimento à esquerda, e a cabeça da fita já se encontra na célula mais à esquerda
  - máquina pára
  - w é rejeitada

◆ Definição formalmente do comportamento

- necessário estender a definição da função programa
- argumento: um estado e uma palavra
- exercício

Def: Linguagens Aceita, Rejeitada, Loop

Linguagem Aceita ou Linguagem Reconhecida por M

ACEITA(M) ou L(M)

- conjunto de todas as palavras de Σ^* aceitas por M, a partir de q_0

Linguagem Rejeitada por M

REJEITA(M)

- conjunto de todas as palavras de Σ^* rejeitadas por M, a partir de q_0

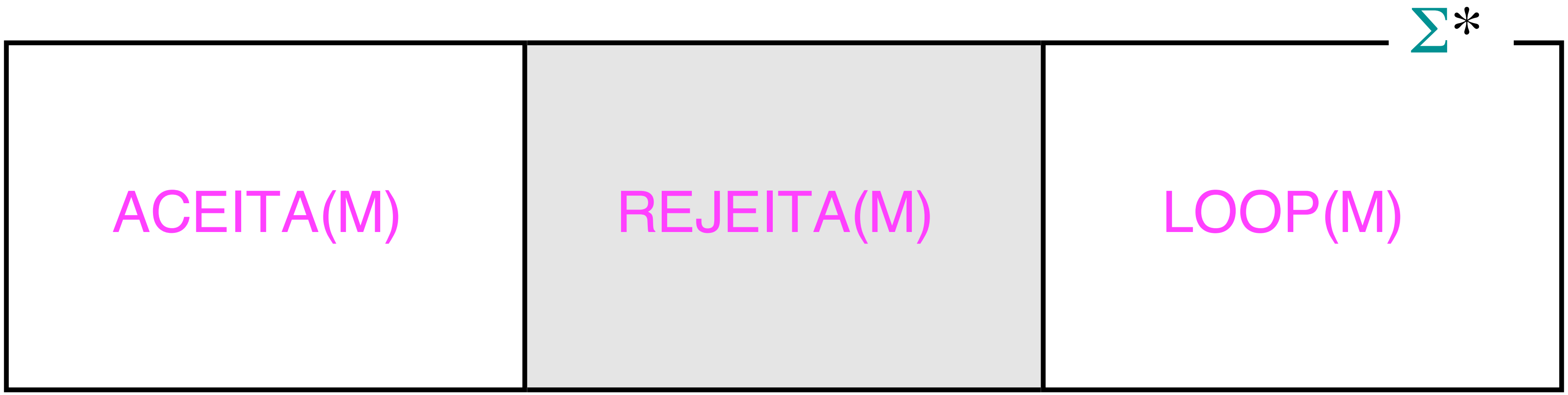
Linguagem Loop de M

LOOP(M)

- conjunto de todas as palavras de Σ^* para as quais M fica processando indefinidamente a partir de q_0

◆ Cada máquina de Turing M sobre Σ

- induz uma partição de Σ^*
- em classes de equivalência
  - ACEITA(M), REJEITA(M) e LOOP(M)
  - se um ou dois dos conjuntos for vazios?



Exp: Máquina de Turing: Duplo Balanceamento

$$L = \{ a^n b^n \mid n \geq 0 \}$$

Máquina de Turing

$$M = (\{ a, b \}, \{ q_0, q_1, q_2, q_3, q_4 \}, \delta, q_0, \{ q_4 \}, \{ A, B \}, \beta, ❂)$$

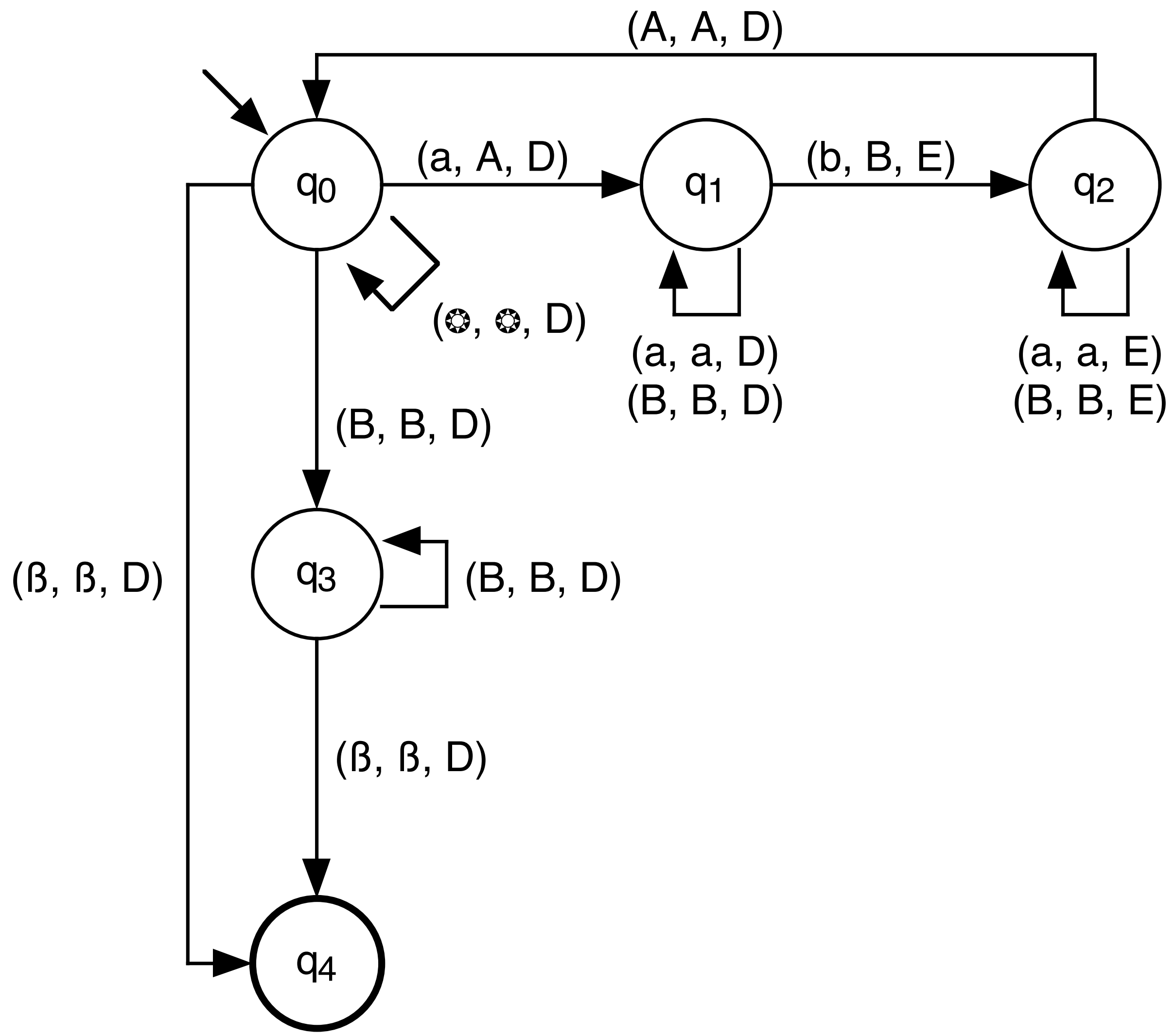
é tal que

$$ACEITA(M) = L \quad e \quad REJEITA(M) = \sim L$$

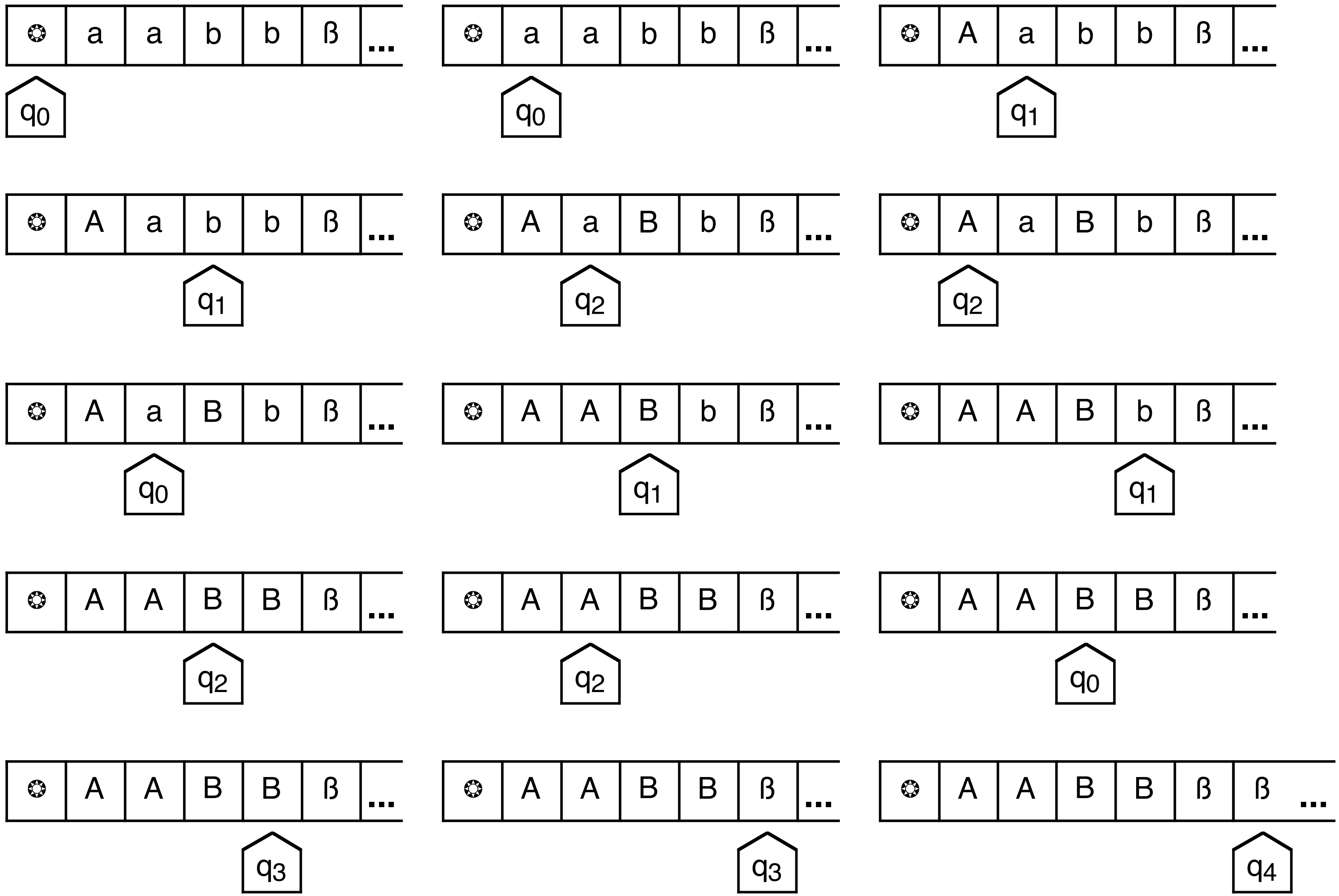
e, portanto, $LOOP(M) = \varnothing$

- qualquer palavra que não esteja na forma $a^x b^x$ é rejeitada

Exp: Máquina de Turing: Duplo Balanceamento



δ	❂	a	b	A	B	β
q_0	$(q_0$, ❂, $D)$	(q_1, A, D)			(q_3, B, D)	(q_4, β, D)
q_1		(q_1, a, D)	(q_2, B, E)		(q_1, B, D)	
q_2		(q_2, a, E)		(q_0, A, D)	(q_2, B, E)	
q_3					(q_3, B, D)	(q_4, β, D)
q_4						


❂ a a b b ß ...
q0
❂ a a b b ß ...
q0
❂ A a b b ß ...
q1
❂ A a b b ß ...
q1
❂ A a B b ß ...
q2
❂ A a B b ß ...
q2
❂ A a B b ß ...
q0
❂ A A B b ß ...
q1
❂ A A B b ß ...
q1
❂ A A B B ß ...
q2
❂ A A B B ß ...
q2
❂ A A B B ß ...
q0
❂ A A B B ß ...
q3
❂ A A B B ß ...
q3
❂ A A B B ß ß ...
q4

Obs: Máquina de Turing × Algoritmo

Foi afirmado que Máquina de Turing

- é aceita como uma formalização do conceito de algoritmo

Entretanto, também é usual considerar como conceito de algoritmo

- máquina de Turing que sempre pára para qualquer entrada

Nesse caso, uma máquina que eventualmente fica em *loop* infinito

- *não* seria considerada um algoritmo

8 - Linguagens Recursivamente Enumeráveis e Sensíveis ao Contexto

8.2 Modelos Equivalentes à Máquina de Turing

- **Uma razão para considerar a máquina de Turing como o mais geral dispositivo de computação**
  - todos os demais modelos e máquinas propostos
  - bem como as diversas modificações da máquina de Turing
  - possuem, no máximo, o mesmo poder computacional da máquina de Turing

◆ Autômato com Múltiplas Pilhas

- autômato com duas pilhas (citado no estudo das LLC)
  - ∗ poder computacional é equivalente ao da máquina de Turing
- maior número de pilhas: não aumenta a capacidade computacional
- exercício
  - ∗ definição formal: autômato com duas (múltiplas) pilhas
  - ∗ equivalência deles ao modelo da máquina de Turing
- como são necessárias duas pilhas, pode-se afirmar
  - ∗ a estrutura de fita é mais expressiva do que a de pilha

◆ Máquina de Turing Não-Derminística

- não aumenta o poder computacional da máquina de Turing

◆ Máquina de Turing com Fita Infinita à Esquerda e à Direita

- fita infinita dos dois lados não aumenta o poder computacional
- pode ser facilmente simulada por uma fita tradicional
  - células pares: parte direita da fita
  - células ímpares: parte esquerda da fita

▪▪▪	a_{-3}	a_{-2}	a_{-1}	a_1	a_2	a_3	▪▪▪

❂	a_1	a_{-1}	a_2	a_{-2}	a_3	a_{-3}	▪▪▪

◆ Máquina de Turing com Múltiplas Fitas

- k fitas infinitas à esquerda e à direita e k cabeças de fita
- função programa
  - dependendo do estado corrente e do símbolo lido em cada fita
  - grava um novo símbolo em cada fita
  - move cada cabeça independentemente
  - assume um (único) novo estado
- inicialmente
  - palavra de entrada: armazenada na primeira fita
  - demais: brancos

◆ Máquina de Turing Multidimensional

- fita tradicional
  - substituída por uma estrutura do tipo arranjo k-dimensional
  - infinita em todas as 2k direções

◆ Máquina de Turing com Múltiplas Cabeças

- k cabeças de leitura e gravação sobre a mesma fita
  - movimentos independentes

- processamento depende
  - estado corrente
  - símbolo lido em cada uma das cabeças

◆ Modificações combinadas sobre a Máquina de Turing

- combinação de algumas ou todas as modificações
  - ∗ não aumenta o poder computacional da máquina de Turing

- exemplo, uma máquina de Turing
  - ∗ não-determinística
  - ∗ com múltiplas fitas
  - ∗ múltiplas cabeças
  - ∗ pode ser simulada por uma máquina de Turing tradicional

8 - Linguagens Recursivamente Enumeráveis e Sensíveis ao Contexto

8.3 Hipótese de Church

- **Objetivo do modelo abstrato de computação Máquina de Turing**
  - explorar os limites da capacidade de expressar soluções de problemas
- **Portanto, uma proposta de**
  - definição formal da noção intuitiva de algoritmo
- **Diversos outros trabalhos: equivalentes ao de Turing**
  - Máquina de Post (Post - 1936)
  - Funções Recursivas (Kleene - 1936)

Forte reforço da Hipótese de (Turing-) Church

"A capacidade de computação representada pela máquina de Turing é o limite máximo que pode ser atingido por qualquer dispositivo de computação".

Em outras palavras

- qualquer outra forma de expressar algoritmos terá, no máximo, a mesma capacidade computacional da máquina de Turing

Hipótese de Church não é demonstrável

- algoritmo ou função computável: noção intuitiva

8 - Linguagens Recursivamente Enumeráveis e Sensíveis ao Contexto

8.4 Máquina de Turing como Reconhecedor

- **Classes de linguagens definidas a partir do formalismo máquina de Turing**
  - Linguagens Recursivamente Enumeráveis
  - Linguagens Recursivas

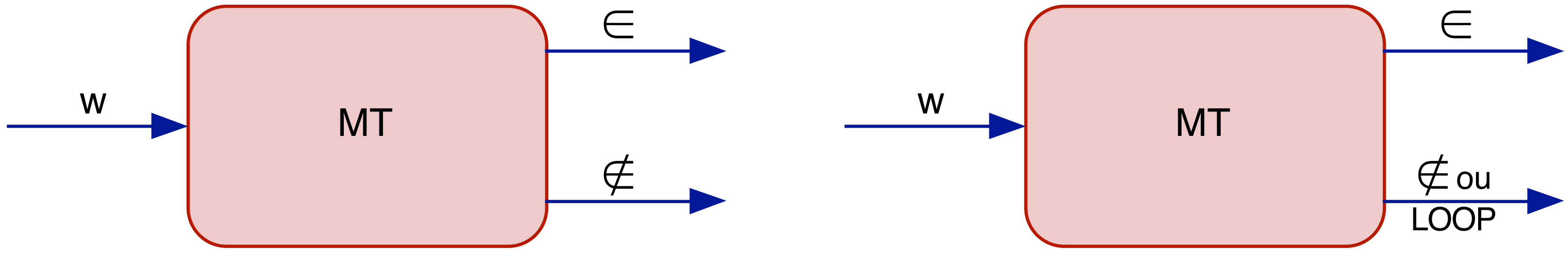
Classe das Linguagens Recursivamente Enumeráveis

- existe uma máquina de Turing capaz de determinar
  - se uma palavra w pertence à linguagem
- entretanto, se $w \in \sim L$, o algoritmo pode
  - *parar*: w não pertence à linguagem
  - *ficar em loop* infinito

Classe das Linguagens Recursivas

- existe pelo menos uma máquina de Turing que *sempre pára*, capaz de determinar se
  - $w \in L$ ou
  - $w \in \sim L$

◆ Recursivas × Recursivamente Enumeráveis



◆ Aparente contradição

reconhecer o complemento de uma linguagem pode ser impossível,
mesmo que seja possível reconhecer a linguagem

8 - Linguagens Recursivamente Enumeráveis e Sensíveis ao Contexto

8.4.1 Linguagem Recursivamente Enumerável

Def: Linguagem Recursivamente Enumerável ou Tipo 0

Uma linguagem aceita por uma máquina de Turing

Exp: Linguagem Recursivamente Enumerável

- $\{ a^n b^n \mid n \geq 0 \}$ já apresentado
- $\{ w \mid w$ tem o mesmo número de símbolos a e $b \}$ exercício
- $\{ a^i b^j c^k \mid i = j$ ou $j = k \}$ exercício

◆ Considerando a Hipótese de Church

- a máquina de Turing é o mais geral dispositivo de computação
- então, a Classe das Linguagens Recursivamente Enumeráveis
  - todas as linguagens que podem ser reconhecidas mecanicamente

◆ Linguagens Recursivamente Enumeráveis × Universo de todas as linguagens

- classe de linguagens muito rica
- entretanto, existem conjuntos que *não* são recursivamente enumeráveis
  - *não* é possível desenvolver uma MT que os reconheça

Teorema: Linguagem Não-Recursivamente Enumerável

Seja $\Sigma = \{ a, b \}$

Suponha X_i o i-ésimo elemento na ordenação lexicográfica de Σ^*

- 0 - ε
- 1 - a
- 2 - b
- 3 - aa
- ...

Exercícios

- é possível codificar todas as máquinas de Turing
- como uma palavra sobre Σ de tal forma que
- cada código represente uma única máquina de Turing
- suponha o conjunto dos códigos ordenados lexicograficamente
- suponha que T_i representa o i-ésimo código nesta ordenação

Então *não* é linguagem recursivamente enumerável

$$L = \{ x_i \mid x_i \text{ não é aceita por } T_i \}$$

<u>Prova:</u> (*por absurdo*)

Suponha que L é recursivamente enumerável

- existe uma máquina de Turing que aceita L
- seja T_k a codificação desta máquina de Turing: $ACEITA(T_k) = L$

Assim

- por definição de L, $x_k \in L$ sse x_k *não* é aceita por T_k
- como T_k aceita L, $x_k \in L$ sse x_k é aceita por T_k

Contradição!!!

Logo, L não é linguagem recursivamente enumerável

Obs: Cardinal dos Problemas > Cardinal dos Algoritmos

Conjunto das condifições de todas as máquinas de Turing

- isomorfo a um subconjunto infinito dos números naturais

Logo, é enumerável (infinitamente contável) o conjunto de todas

- máquinas de Turing
- linguagens recursivamente enumeráveis
- problemas solucionáveis

Em contrapartida, o conjunto das linguagens que *não* são recursivamente enumeráveis (problemas *não*-solucionáveis)

- não-contável

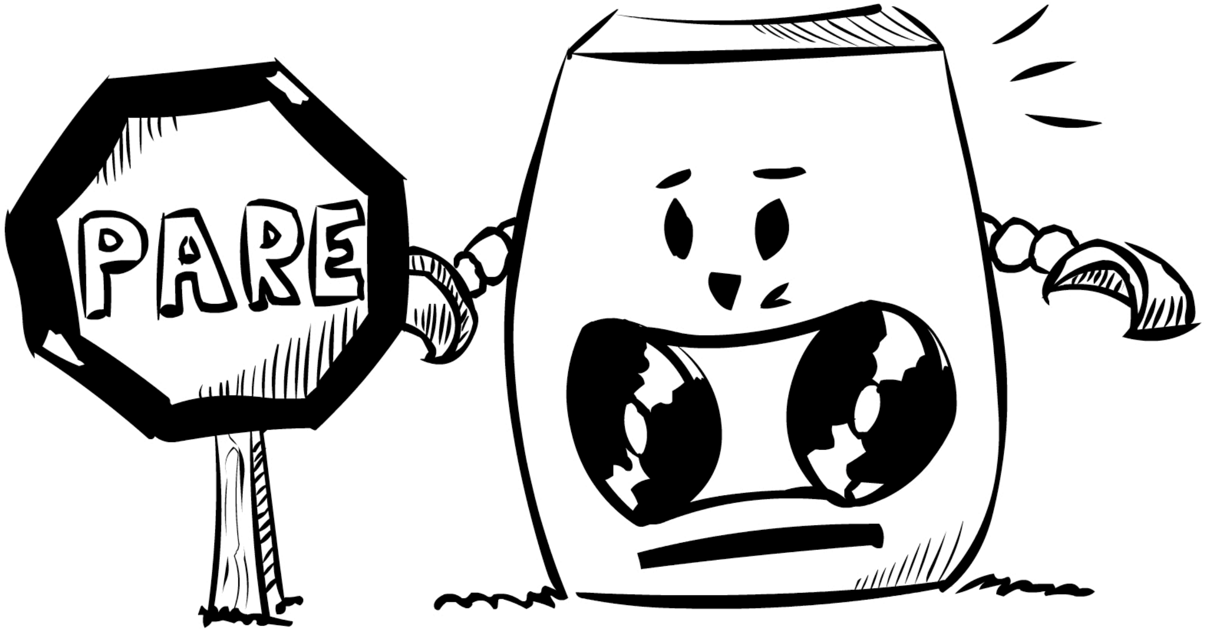
Portanto, computacionalmente

existem mais problemas do que algoritmos para resolvê-los.

Exemplo

$$\{ f: \mathbf{N} \to \mathbf{N} \mid f \text{ é função} \}$$

- classe muito particular de problemas (linguagens)
- prova-se: isomorfo a $\mathbf{R}$ (cardinal é $2^{\aleph_0}$)
- maior do que $\aleph_0$ (cardinal do conjunto das máquinas de Turing)



8.1.2 Linguagem Recursiva

Def: Linguagem Recursiva

Existe pelo menos uma máquina de Turing M

- ACEITA(M) = L
- REJEITA(M) = ~L

Exp: Linguagem Recursiva

- $\{ a^n b^n \mid n \geq 0 \}$
- $\{ a^n b^n c^n \mid n \geq 0 \}$
- $\{ w \mid w \in \{ a, b \}^*$ e tem o dobro de símbolos a que $b \}$

8 - Linguagens Recursivamente Enumeráveis e Sensíveis ao Contexto

8.5 Propriedades das Linguagens Recursivamente Enumeráveis e Recursivas

◆ Algumas das principais propriedades

- complemento de uma linguagem recursiva é recursiva
- linguagem é recursiva sse a linguagem e seu complemento são linguagens recursivamente enumeráveis
- Classe das Linguagens Recursivas está contida propriamente na Classe das Linguagens Recursivamente Enumeráveis

Teorema: Complemento de uma Linguagem Recursiva é Recursiva

Se uma linguagem L sobre um alfabeto Σ qualquer é recursiva, então o seu complemento ~L é recursiva

Prova: (*direta*)

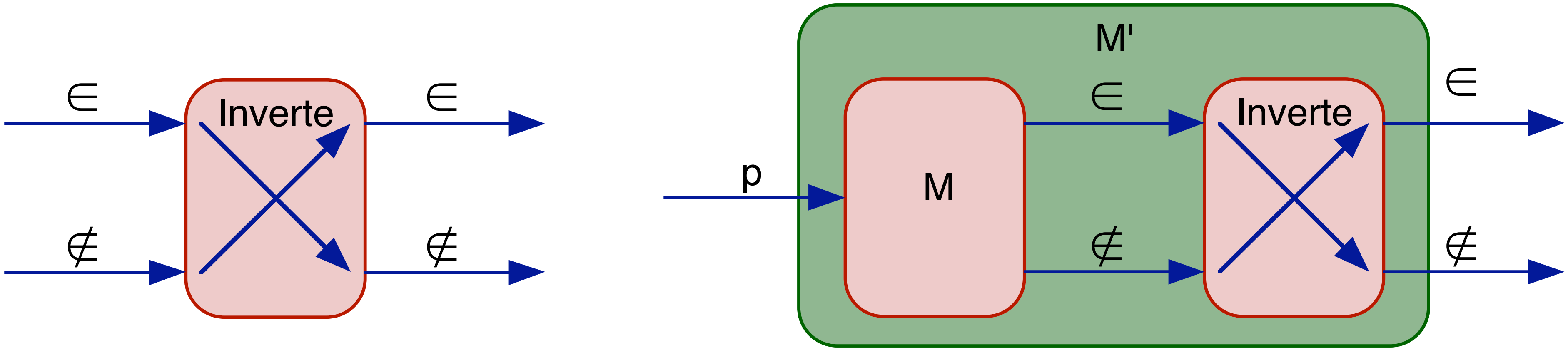
Suponha L linguagem recursiva. Então existe M, máquina de Turing

- ACEITA(M) = L
- REJEITA(M) = ~L
- LOOP(M) = ∅

Seja

* Inverte uma máquina de Turing que inverte ACEITA / REJEITA
* M’ máquina de Turing resultante da composição de Inverte e M
  * M’ aceita a linguagem ~L
  * sempre pára para qualquer entrada

Portanto, o complemento de uma linguagem recursiva é recursiva



Teorema: Linguagem Recursiva × Recursivamente Enumerável

L é recursiva sse L e ~L são recursivamente enumeráveis

Prova:

($\Rightarrow$ *direta*)

Suponha L linguagem recursiva. Então (teorema anterior)

- ~L é recursiva

Como toda linguagem recursiva também é recursivamente enumerável

- L e ~L são recursivamente enumeráveis

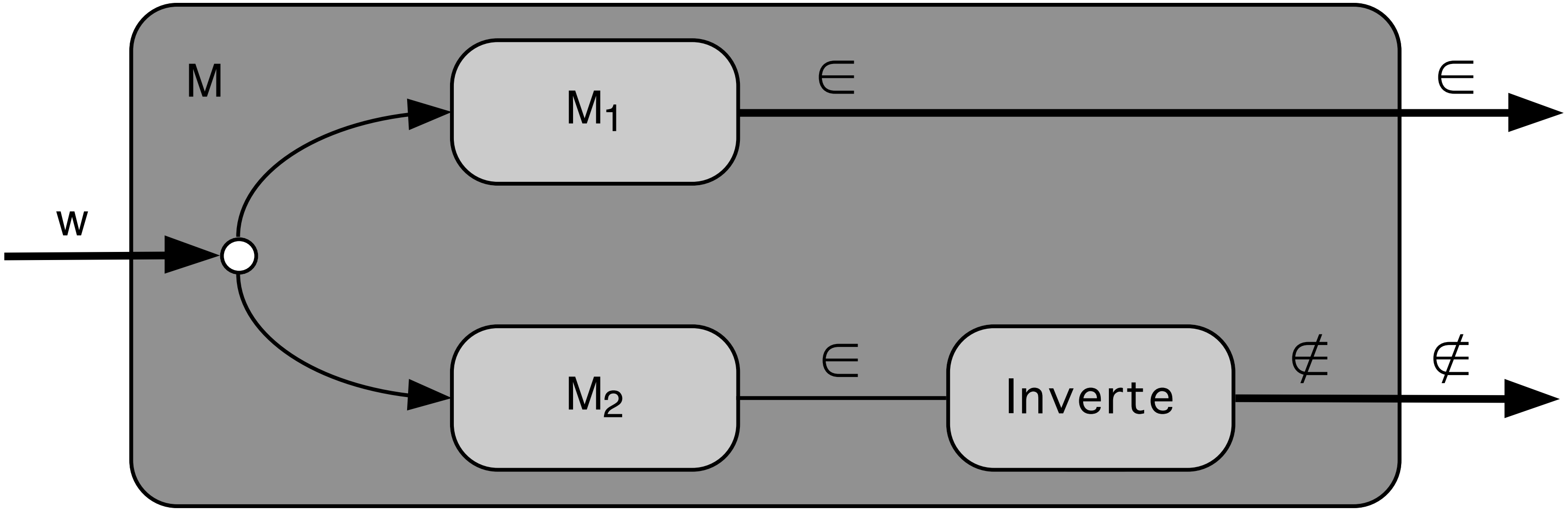
($\Leftarrow$ *direta*)

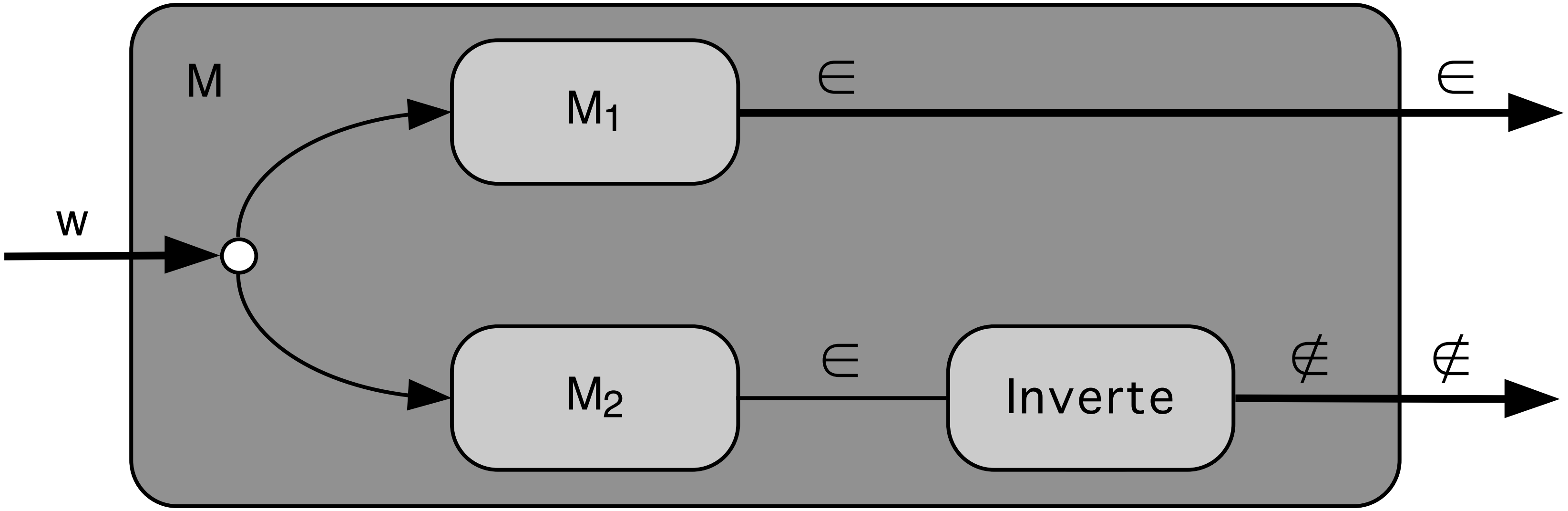
Suponha L linguagem tal que L e ~L são recursivamente enumeráveis

Então existem M_1 e M_2, máquinas de Turing, tais que

- ACEITA(M_1) = L
- ACEITA(M_2) = ~L

Seja M máquina de Turing resultante da composição





- composição não-determinista de M_1 com M_2
- composição seqüencial de M_1 com Inverte

Para qualquer palavra de entrada

- M aceita se M_1 aceita
- M rejeita se M_2 aceita

Logo, L é recursiva

Teorema: Linguagens Recursivas $\subset$ Recursivamente Enumeráveis

Prova: (*direta*)

Mostrar inclusão própria

- existe pelos menos uma linguagem recursivamente enumerável que *não* é recursiva

Linguagem recursivamente enumerável que é não-recursiva

$$L = \{ X_i \mid X_i \text{ é aceita por } T_i \}$$

L é Recursivamente Enumerável (esboço da máquina de Turing)

- M gera palavras X_1, X_2,... em ordem lexicográfica
  - ∗ compara com w
  - ∗ quando $X_i = w$, w é a i-ésima palavra na enumeração
- M gera T_i, a i-ésima máquina de Turing (exercícios)
- M simula T_i para a entrada $w = X_i$
  - ∗ se w pertence a $ACEITA(T_i)$, então w pertence a $ACEITA(M)$
  - ∗ simulador: exercício
- M aceita w sse $X_i = w$ é aceita por T_i

Logo, L é recursivamente enumerável

L não é Recursiva

Já foi visto

- L é recursiva sse L e ~L são recursivamente enumeráveis

Complemento de L *não* é recursivamente enumerável

- então L é *não*-recursiva

8 - Linguagens Recursivamente Enumeráveis e Sensíveis ao Contexto

8.6 Gramática Irrestrita

◆ Gramática Irrestrita

- gramática *sem* qualquer restrição nas produções

Exp: Gramática Irrestrita: $\{ a^n b^n c^n \mid n \geq 0 \}$

$$G = ???$$

Exp: Gramática Irrestrita: $\{ a^n b^n c^n \mid n \geq 0 \}$

$$G = (\{ S, C \}, \{ a, b, c \}, P, S)$$

- $P = \{ S \to abc \mid \varepsilon, ab \to aabbC, Cb \to bC, Cc \to cc \}$

Derivação de aaabbbccc

$$S \Rightarrow abc \Rightarrow aabbCc \Rightarrow aaabbCbCc \Rightarrow$$
$$aaabbCbcc \Rightarrow aaabbbCcc \Rightarrow aaabbbccc$$

- C "caminha" na palavra até a posição correta para gerar c

Teorema: Linguagem Recursivamente Enumerável × Gramática Irrestrita

L é linguagem recursivamente enumerável sse L é gerada por uma gramática irrestrita

- (não será demonstrado)

8 - Linguagens Recursivamente Enumeráveis e Sensíveis ao Contexto

8.7 Linguagem Sensível ao Contexto

Def: Gramática Sensível ao Contexto

$$G = (V, T, P, S)$$

Qualquer regra de produção de P é da forma $\alpha \to \beta$

- β é palavra de $(V \cup T)^*$
- α é palavra de $(V \cup T)^+$ tal que $|\alpha| \leq |\beta|$
  - excetuando-se, eventualmente, para $S \to \varepsilon$
  - neste caso, S não está no lado direito de qualquer produção

- **Portanto, em uma gramática sensível ao contexto**
  - a cada etapa de derivação
  - tamanho da palavra derivada *não* pode diminuir
    - excetuando-se para gerar a palavra vazia
    - se esta pertencer à linguagem
- **Observe (por quê?)**
  - *nem* toda gramática livre do contexto é sensível ao contexto

Def: Linguagem Sensível ao Contexto, Linguagem Tipo 1

Linguagem gerada por uma gramática sensível ao contexto

Exp: Linguagem Sensível ao Contexto: Palavra Duplicada

$$\{ ww \mid w \text{ é palavra de } \{ a, b \}^* \}$$

$$G = (\{ S, X, Y, A, B, \langle aa \rangle, \langle ab \rangle, \langle ba \rangle, \langle bb \rangle \}, \{ a, b \}, P, S)$$

Produções de P

- $S \rightarrow XY \mid aa \mid bb \mid \varepsilon$,
- $X \rightarrow XaA \mid XbB \mid aa\langle aa \rangle \mid ab\langle ab \rangle \mid ba\langle ba \rangle \mid bb\langle bb \rangle$,
- $Aa \rightarrow aA$, $Ab \rightarrow bA$, $AY \rightarrow Ya$,
- $Ba \rightarrow aB$, $Bb \rightarrow bB$, $BY \rightarrow Yb$,
- $\langle aa \rangle a \rightarrow a\langle aa \rangle$, $\langle aa \rangle b \rightarrow b\langle aa \rangle$, $\langle aa \rangle Y \rightarrow aa$,
- $\langle ab \rangle a \rightarrow a\langle ab \rangle$, $\langle ab \rangle b \rightarrow b\langle ab \rangle$, $\langle ab \rangle Y \rightarrow ab$,
- $\langle ba \rangle a \rightarrow a\langle ba \rangle$, $\langle ba \rangle b \rightarrow b\langle ba \rangle$, $\langle ba \rangle Y \rightarrow ba$,
- $\langle bb \rangle a \rightarrow a\langle bb \rangle$, $\langle bb \rangle b \rightarrow b\langle bb \rangle$, $\langle bb \rangle Y \rightarrow bb$

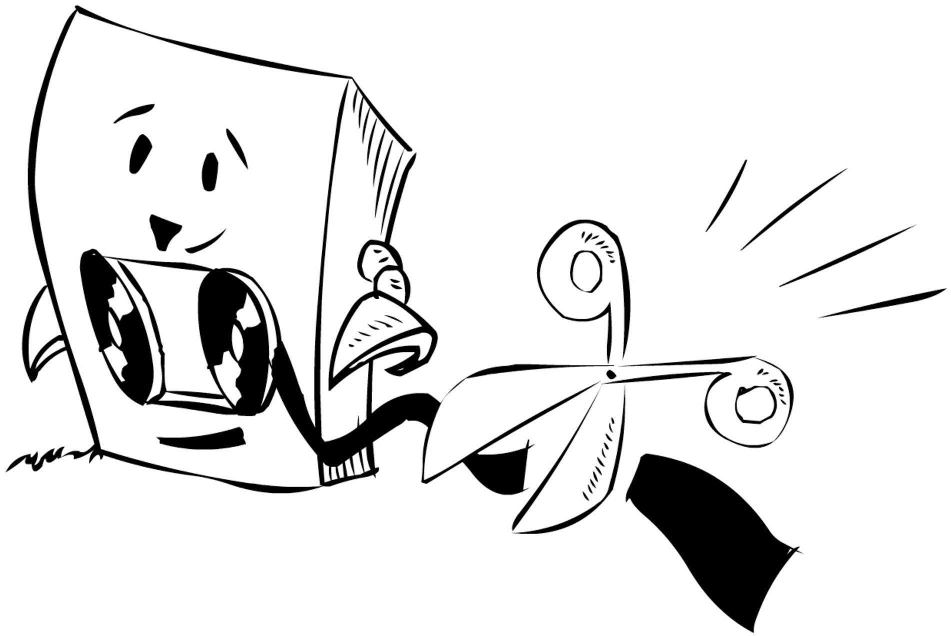
Gera o primeiro w após X, e o segundo w após Y

- a cada terminal gerado após X
  - gerada correspondente variável
- variável "caminha" na palavra até passar por Y
  - deriva o correspondente terminal

- para encerrar
  - X deriva subpalavra de dois terminais
  - e correspondente variável a qual "caminha" até encontrar Y
  - quando é derivada a mesma subpalavra de dois terminais

- se X derivar uma subpalavra de somente um terminal (e a correspondente variável) ?

8 - Linguagens Recursivamente Enumeráveis e Sensíveis ao Contexto



8.8 Máquina de Turing com Fita Limitada

Máquina de Turing com Fita Limitada

- máquina de Turing
- fita limitada ao tamanho da entrada
- mais duas células
  - marcadores de início e de fim de fita

Não-Determinismo?

- não é conhecido se aumenta o poder computacional

Def: Máquina de Turing com Fita Limitada (MTFL)

$$M = (\Sigma, Q, \delta, q_0, F, V, ✪, †)$$

- Σ - alfabeto (de símbolos) de entrada
- Q - conjunto de estados (finito)
- δ - (função) programa ou função de transição (parcial)

$$\delta: Q \times (\Sigma \cup V \cup \{ ✪, † \}) \rightarrow 2^{Q \times (\Sigma \cup V \cup \{✪, †\}) \times \{E,D\}}$$

∗ transição: $\delta(p, x) = \{ (q_1, y_1, m_1), \ldots, (q_n, y_n, m_n) \}$

- q_0 - estado inicial: elemento distinguido de Q
- F - conjunto de estados finais: subconjunto de Q
- V - alfabeto auxiliar (pode ser vazio)
- ✪ - símbolo de início ou marcador de início da fita
- † - símbolo de fim ou marcador de fim da fita

Exp: Máquina de Turing com Fita Limitada: Palavra Duplicada

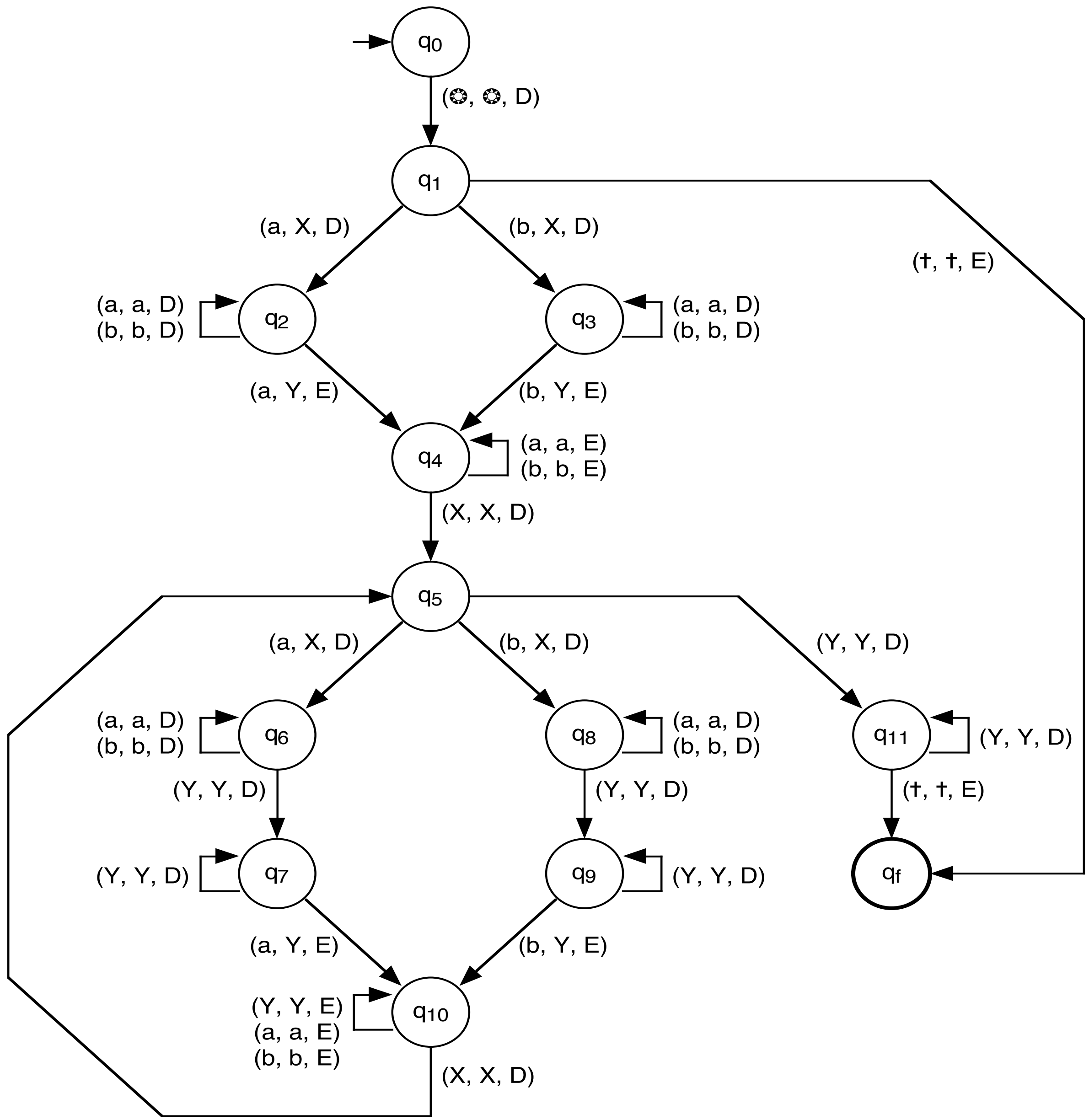
$$L = \{ ww \mid w \text{ é palavra de } \{ a, b \}^* \}$$

A máquina de Turing com fita limitada

$$M = (\{ a, b \}, \{ q_0, q_1, \ldots, q_9, q_f \}, \delta, q_0, \{ q_f \}, \{ X, Y \}, ❂, †)$$

é tal que $ACEITA(M) = L$ e $REJEITA(M) = \sim L$

- q_1, o início do primeiro w é marcado com um X
- q_2 e q_3 definem não-determinismos
  - marcar com um Y o início do segundo w
- q_5 a q_{11} verifica a igualdade das duas metades


q0
(⚪, ⚪, D)
q1
(a, X, D)
(b, X, D)
(†, †, E)
(a, a, D)
(b, b, D)
q2
q3
(a, a, D)
(b, b, D)
(a, Y, E)
(b, Y, E)
q4
(a, a, E)
(b, b, E)
(X, X, D)
q5
(a, X, D)
(b, X, D)
(Y, Y, D)
(a, a, D)
(b, b, D)
q6
q8
(a, a, D)
(b, b, D)
q11
(Y, Y, D)
(Y, Y, D)
(Y, Y, D)
(†, †, E)
(Y, Y, D)
q7
q9
(Y, Y, D)
qf
(a, Y, E)
(b, Y, E)
(Y, Y, E)
(a, a, E)
(b, b, E)
q10
(X, X, D)

Teorema: Linguagem Sensível ao Contexto $\times$ Máquina de Turing com Fita Limitada

L é uma linguagem sensível ao contexto sse

L é reconhecida por uma máquina de Turing com fita limitada

- não será demonstrado

Linguagens Formais e Autômatos

P. Blauth Menezes

Linguagens Formais e Autômatos

P. Blauth Menezes

blauth@inf.ufrgs.br

Departamento de Informática Teórica

Instituto de Informática / UFRGS