Jhonathan de Moura Santos

RGM: 32813589

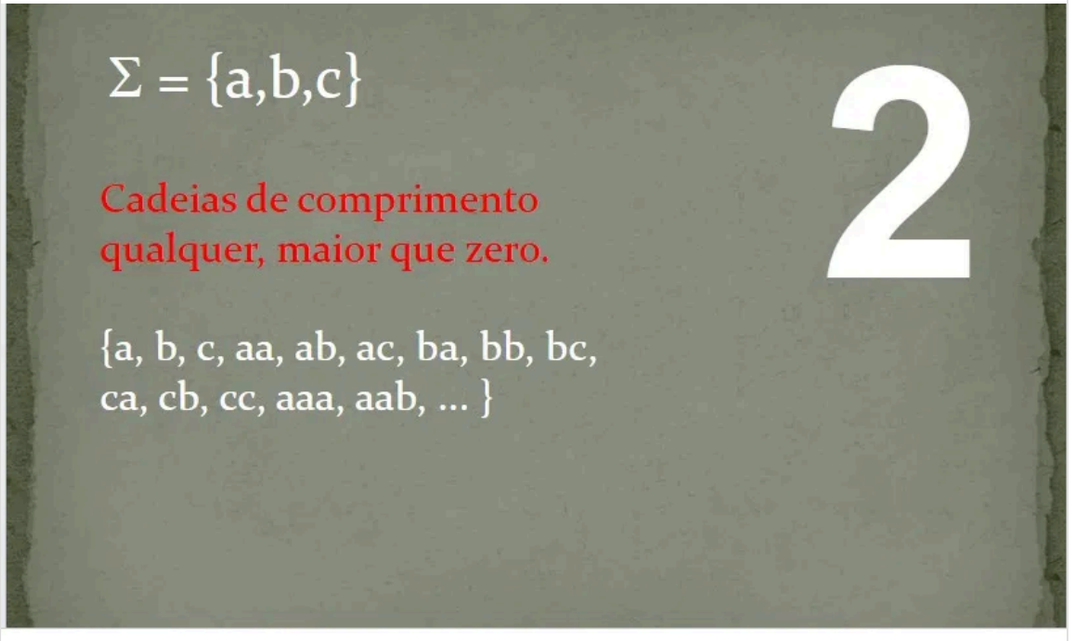
Linguagem Formal e Autômatos

Linguagem Formal e Autômatos

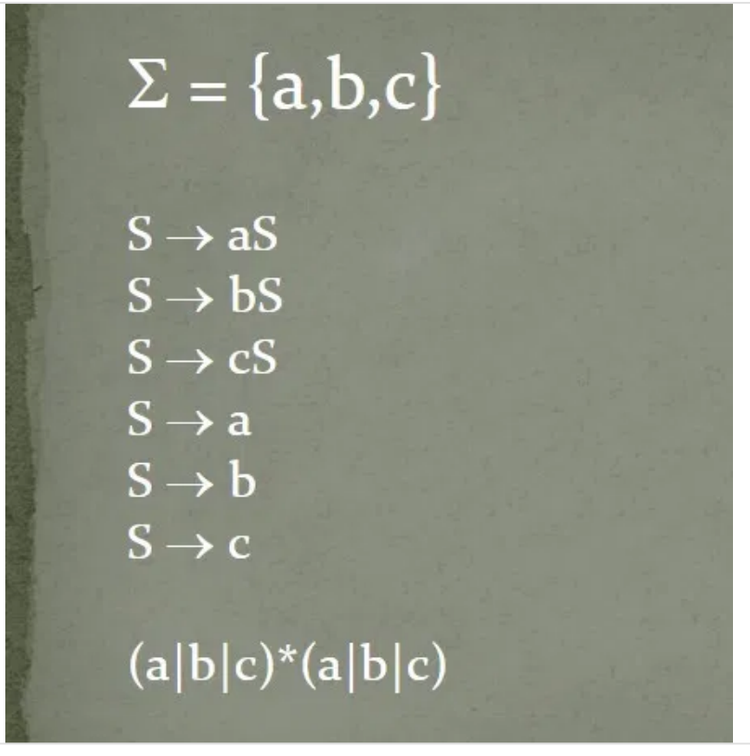
Autômato Finito Determinístico para Reconhecimento de Cadeias Não-Vazias

Referências Teóricas

Alfabeto e Linguagem



Gramática da Linguagem



Estas imagens apresentam a base teórica da linguagem formal que estamos estudando:

- À esquerda: Definição do alfabeto Σ = {a,b,c} e exemplos de cadeias aceitas pela linguagem
- À direita: Representação da gramática que gera a linguagem e sua expressão regular equivalente (a|b|c)*(a|b|c)

Estas definições formais são a base para o autômato finito determinístico que implementaremos.

Alfabeto e Linguagem

Alfabeto: $\Sigma = \{a, b, c\}$
Linguagem: Todas as cadeias não vazias sobre Σ (ou seja, qualquer string formada por a, b ou c, com comprimento ≥ 1).

Exemplos de cadeias aceitas:

- a
- b
- c
- ab
- cba
- aaabcc

Cadeia rejeitada:

- Apenas a cadeia vazia ("")

Esta linguagem representa o conjunto de todas as sequências possíveis formadas pelos símbolos a, b e c, desde que tenham pelo menos um símbolo.

Modelo do Autômato Finito Determinístico (AFD)

Estados:	`{q0, q1}`
Alfabeto:	`{a, b, c}`
Estado inicial:	`q0`
Estado de aceitação:	`{q1}`

Função de Transição:

- → $\delta(q0, x) = q1$, para $x \in \{a, b, c\}$
- → $\delta(q1, x) = q1$, para $x \in \{a, b, c\}$

Implementação em Python

```python
# Autômato finito que reconhece cadeias sobre {a, b, c} de comprimento >= 1

# Definição do autômato:
# Estados = {q0 (inicial), q1 (aceitação)}
# Alfabeto = {a, b, c}
# Função de transição:
#   δ(q0, x) = q1, para x ∈ {a, b, c}
#   δ(q1, x) = q1, para x ∈ {a, b, c}
# Estado inicial = q0
# Estado de aceitação = {q1}

def validar_cadeia(cadeia: str) -> bool:
    estado = "q0"

    # Se a cadeia for vazia, rejeita imediatamente
    if cadeia == "":
        return False

    for simbolo in cadeia:
        if simbolo not in {"a", "b", "c"}:
            # Se encontrar símbolo fora do alfabeto, rejeita
            return False

        if estado == "q0":
            estado = "q1"  # ao ler o primeiro símbolo, vai para estado de aceitação
        elif estado == "q1":
            estado = "q1"  # permanece em aceitação

    return estado == "q1"


# ------------------------------
# Entrada das cadeias para validação
while True:
    cadeia = (input("Digite uma cadeia (ou 'sair' para encerrar): ")).strip()

    if cadeia.lower() == "sair":
        print("Encerrando o validador. Até mais!")
        break

    if validar_cadeia(cadeia):
        print(f"Cadeia '{cadeia}' -> ACEITA ✅\n")
    else:
        print(f"Cadeia '{cadeia}' -> REJEITA ❌\n")
```

> O código acima implementa um validador de cadeias baseado no AFD definido anteriormente. A função **validar_cadeia()** simula o comportamento do autômato, verificando se uma cadeia de entrada é aceita ou rejeitada.

1. Inicializa no estado q0 e rejeita imediatamente se a cadeia for vazia
2. Verifica se cada símbolo pertence ao alfabeto {a, b, c}
3. Implementa as transições: q0 → q1 ao ler o primeiro símbolo, e q1 → q1 para os demais
4. Aceita a cadeia se terminar no estado de aceitação q1

Exemplos de Validação

Cadeias Aceitas

`"a"`	✅ ACEITA
`"bc"`	✅ ACEITA
`"abc"`	✅ ACEITA
`"aaabbbccc"`	✅ ACEITA

Cadeias Rejeitadas

`""`	❌ REJEITA
`"d"`	❌ REJEITA
`"ab2c"`	❌ REJEITA

Funcionamento do autômato: O autômato inicia no estado q0 e, ao ler qualquer símbolo do alfabeto (a, b ou c), passa para o estado de aceitação q1. Uma vez em q1, permanece neste estado para qualquer símbolo subsequente do alfabeto. A cadeia vazia é rejeitada porque não há transição que permita aceitar sem ler símbolos. Símbolos fora do alfabeto também causam rejeição.