

Unidade 1: Fundamentos Matemáticos, Alfabetos e Linguagens

1. (Modelagem e Teoria dos Conjunto)

Na Teoria da Computação, os modelos formais abstratos antecedem a existência do hardware físico para garantir a construção de algoritmos sem ambiguidades semânticas. O conceito de Conjunto das Partes $P(A)$, por exemplo, é a base matemática para descrever o estado de incerteza/paralelismo nos autômatos. Se um determinado alfabeto de entrada é definido como $\Sigma = \{a, b, c\}$, a cardinalidade exata e o número de elementos contidos em seu Conjunto das Partes $P(\Sigma)$ correspondem a:

- a) 3 elementos, representados apenas pelos caracteres unitários $\{a\}$, $\{b\}$, $\{c\}$.
- b) 6 elementos, equivalentes ao produto cartesiano simples $\Sigma \times \Sigma$.
- c) 8 elementos, englobando a combinação do conjunto vazio $\emptyset$, subconjuntos unitários, pares e o conjunto total Σ .
- d) 9 elementos, calculados através da potência quadrática $|\Sigma|^2$.
- e) Infinitos elementos, devido à propriedade do Fechamento de Kleene Σ^* .

2. (Cadeia Vazia e Elemento Neutro)

Na manipulação de dados textuais e análise sintática, a palavra vazia (ϵ) desempenha um papel rigorosamente definido. Ao analisar as propriedades de comprimento e concatenação de ϵ , assinale a afirmativa correta:

- a) O comprimento de ϵ é $|\epsilon| = 1$, pois ocupa uma célula de memória na fita e altera o tamanho final da string.
- b) O comprimento de ϵ é $|\epsilon| = 0$, atuando como o elemento neutro na operação de concatenação, onde $w \cdot \epsilon = w$ e $|\epsilon| = |\epsilon| + 0 = |\epsilon|$.
- c) A concatenação de uma palavra w com a palavra vazia ϵ resulta obrigatoriamente na palavra vazia ϵ .
- d) A palavra vazia pertence exclusivamente às Linguagens Livres de Contexto, sendo proibida nas Linguagens Regulares.
- e) A operação de inversão do reverso em uma palavra vazia (ϵ^r) gera um erro de execução sintático no autômato.

3. (Operadores de Linguagens: Fechamento de Kleene vs. Positivo)

Considere um alfabeto binário $\Sigma = \{0, 1\}$. Na formalização matemática das linguagens, a diferença fundamental entre o Fechamento de Kleene (Σ^*) e o Fechamento Positivo (Σ^+) reside no fato de que:

- a) Σ^* gera um conjunto finito de cadeias, enquanto Σ^+ gera um conjunto infinito.
- b) Σ^+ inclui obrigatoriamente a cadeia vazia (ϵ), enquanto Σ^* a exclui das suas combinações.
- c) Σ^* inclui a cadeia vazia (ϵ) em seu conjunto, enquanto Σ^+ diferencia-se por exigir palavras de comprimento mínimo $|\sigma| \geq 1$, excluindo ϵ .
- d) Ambos geram conjuntos exatamente idênticos, variando apenas na notação usada pelo simulador JFLAP.
- e) Σ^+ aceita apenas números ímpares de bits, enquanto Σ^* aceita apenas números pares.

4. (Hierarquia de Chomsky e Tipos de Linguagens)

Ao estudar a Hierarquia de Chomsky (Unidade 1), o engenheiro de software percebe que os modelos de máquinas e as linguagens estão organizados por níveis concêntricos de restrição e memória. Correlacione o tipo de linguagem com seu respectivo modelo de automação reconhecedor:

- 1. Linguagens Regulares (Tipo 3)
- 2. Linguagens Livres de Contexto (Tipo 2)
- 3. Linguagens Sensíveis ao Contexto (Tipo 1)
- 4. Linguagens Recursivamente Enumeráveis (Tipo 0)

- () Autômato de Pilha
- () Máquina de Turing
- () Autômato Finito (AFD/AFN)
- () Autômato Linearmente Limitado (LBA)

A sequência correta desse preenchimento é:

- a) 2 - 4 - 1 - 3
- b) 1 - 2 - 3 - 4
- c) 3 - 1 - 4 - 2
- d) 2 - 3 - 1 - 4
- e) 4 - 2 - 1 - 3

5. (Aplicações Práticas da Teoria de Linguagens)

Um compilador moderno divide o processamento do código-fonte em fases bem delimitadas. A validação de nomes de variáveis que começam com letras e podem conter dígitos (como x0 ou taxa1), bem como a busca de tokens rígidos em caixas de texto, utiliza internamente qual classe da Hierarquia de Chomsky?

- a) Linguagens Sensíveis ao Contexto (Tipo 1), através de testes semânticos no sistema operacional.
- b) Linguagens Regulares (Tipo 3), processadas via Autômatos Finitos e Expressões Regulares em tempo real.
- c) Linguagens Livres de Contexto (Tipo 2), que necessitam de pilha para verificar se a variável é do tipo inteiro.
- d) Linguagens Recursivamente Enumeráveis (Tipo 0), executadas por Máquinas de Turing com fita infinita.
- e) Linguagens Não-Formais, baseadas em regras de processamento natural de texto.

Unidade 2: Autômatos Finitos Determinísticos (AFD)

6. (Limitação de Memória nos AFDs)

Em termos de engenharia de computadores, o Autômato Finito Determinístico (AFD) representa a unidade de decisão mais simples de processamento. A principal limitação física e estrutural de um AFD que o impede de resolver problemas complexos como o balanceamento de parênteses aninhados é:

- a) A incapacidade de processar cadeias contendo caracteres alfabéticos ou decimais.
- b) O fato de sua memória ser estritamente limitada ao número fixo de seus estados internos, não possuindo fita de escrita nem pilha de suporte para contagem infinita.
- c) A obrigatoriedade de que a fita de entrada seja apagada a cada transição executada.
- d) A ausência de uma função de transição matemática bem definida no formalismo.
- e) A exigência de que todos os seus estados sejam obrigatoriamente estados finais de aceitação.

7. (Definição Formal do AFD)

A definição formal de um Autômato Finito Determinístico tradicional na literatura de computação é expressa por uma Quintupla $M = (Q, \Sigma, \delta, q_0, F)$. No contexto do determinismo estrito de um AFD:

- I. O conjunto de estados iniciais pode conter mais de um elemento para cobrir falhas de sistema.
- II. A função de transição é um mapeamento $\delta: Q \times \Sigma \rightarrow Q$, associando a cada par (estado atual, símbolo lido) exatamente um único estado destino.
- III. O conjunto de estados de aceitação F deve ser obrigatoriamente um subconjunto de Q ($F \subseteq Q$).

É correto o que se afirma em:

- a) I, apenas.
- b) II, apenas.
- c) I e III, apenas.
- d) II e III, apenas.
- e) I, II e III.

8. (Interpretação de Tabelas de Transição de AFD)

Considere o seguinte AFD modelado para a verificação de paridade de zeros sobre o alfabeto $\Sigma = \{0, 1\}$:

- Estado inicial e de aceitação: $\rightarrow q_{\text{par}}$
- Estado intermediário: q_{impar}
- Transições: $\delta(q_{\text{par}}, 0) = q_{\text{impar}}$, $\delta(q_{\text{par}}, 1) = q_{\text{par}}$, $\delta(q_{\text{impar}}, 0) = q_{\text{par}}$, $\delta(q_{\text{impar}}, 1) = q_{\text{impar}}$.

Ao submeter a cadeia de entrada $w = 1111$ ao autômato, o veredito do simulador (como o JFLAP) e a justificativa técnica para esse resultado serão:

- a) Rejeitada, pois a cadeia não possui nenhum bit 0 em sua composição.
- b) Aceita, pois o número de zeros na cadeia é igual a 0, e matematicamente o número zero é classificado como um número par ($0 \bmod 2 = 0$), mantendo o AFD em seu estado inicial de aceitação q_{par} .
- c) Rejeitada, porque a leitura do bit 1 provoca um erro e direciona a máquina para o estado morto.
- d) Aceita, apenas se a cadeia vazia (ϵ) for concatenada ao final do processamento.
- e) Indeterminada, visto que o autômato entra em um ciclo de loop infinito sem parar.

9. (Função de Transição Estendida δ^*)

Para analisar o comportamento global de um AFD diante de uma palavra inteira $w \in \Sigma^*$ em vez de processar caractere por caractere de forma isolada, a literatura matemática utiliza a extensão recursiva da função de transição denominada:

- a) Função de Transição Nula (δ_0).
- b) Função de Transição Estendida (δ^*).

- c) Matriz de Transição Paralela (δ^p).
- d) Fecho-Vazio (ϵ -closure).
- e) Produto Cartesiano Reverso.

10. (Inclusão do Estado Morto / Dead State)

Durante o projeto de sistemas de segurança (como validação de formatos de arquivos ou senhas), quando uma cadeia de entrada viola irreversivelmente a regra sintática estabelecida no início da leitura, para onde a função de transição δ de um AFD rigoroso deve direcionar o fluxo de processamento?

- a) Para o estado inicial q_0 , reiniciando a leitura do zero.
- b) Para um "Estado Morto" (ou estado de erro/lixreira), de onde nenhuma transição posterior por qualquer símbolo consegue sair, garantindo a rejeição final da cadeia.
- c) Para o conjunto de estados finais F , aceitando a cadeia por exceção de segurança.
- d) Para uma transição nula (ϵ), pulando o símbolo incorreto sem consumi-lo.
- e) O autômato deve ser desligado imediatamente sem gerar resposta.

Unidade 3: Autômatos Finitos Não-Determinísticos (AFN) e Transições- ϵ

11. (Natureza e Assinatura da Função de Transição no AFN)

O Autômato Finito Não-Determinístico (AFN) relaxa as restrições formais do AFD, permitindo maior flexibilidade na modelagem de sistemas. Matematicamente, a função de transição δ de um AFN sobre um conjunto de estados Q e alfabeto Σ é definida como:

- a) $\delta: Q \times \Sigma \rightarrow Q$.
- b) $\delta: Q \times \Sigma \rightarrow P(Q)$, indicando que o resultado de uma transição é um elemento do Conjunto das Partes de Q (um subconjunto de estados possíveis).
- c) $\delta: P(Q) \times Q \rightarrow \Sigma$.
- d) $\delta: \Sigma \times \Gamma \rightarrow F$.
- e) $\delta: Q \times \epsilon \rightarrow Q$.

12. (Múltiplos Caminhos e Critério de Aceitação no AFN)

Durante a simulação de uma cadeia de entrada w em um AFN, a máquina explora múltiplos caminhos computacionais em paralelo (ou realiza "palpites" computacionais). Para que uma cadeia w seja considerada oficialmente aceita por um AFN, é necessário que:

- a) Todos os caminhos computacionais explorados terminem simultaneamente em estados de aceitação.
- b) Pelo menos um dos caminhos computacionais possíveis chegue ao final da leitura da cadeia em um estado pertencente ao conjunto de estados de aceitação F .
- c) O autômato não utilize nenhuma transição por conjunto vazio ($\emptyset$) durante a execução.
- d) O número total de estados visitados seja estritamente igual ao comprimento da string $|w|$.
- e) A cadeia w seja convertida em binário antes do processamento.

13. (Papel Didático e Operacional do Fecho- ϵ)

Em modelos de AFN com transições vazias (AFN- ϵ), o conceito de Fecho- ϵ (ϵ -closure) de um estado q é definido como:

- a) O conjunto de todas as cadeias de texto que fazem o autômato entrar em estado de erro.
- b) O conjunto contendo o próprio estado q acrescido de todos os estados que podem ser alcançados a partir dele realizando exclusivamente transições vazias (ϵ), sem consumir nenhum símbolo da fita.
- c) A remoção de todas as setas do diagrama para simplificar a conversão para AFD.
- d) O número de vezes que a palavra vazia aparece na fita de leitura.
- e) A interseção entre o alfabeto de entrada Σ e o conjunto de estados finais F .

14. (Equivalência Matemática entre AFD e AFN)

Ao comparar a capacidade de reconhecimento de linguagens entre Autômatos Finitos Determinísticos (AFD) e Não-Determinísticos (AFN), o Algoritmo de Construção de Subconjuntos prova conclusivamente que:

- a) O AFN é estritamente mais poderoso que o AFD, sendo capaz de reconhecer linguagens de Tipo 2 da Hierarquia de Chomsky.
- b) O AFD é mais poderoso que o AFN, pois possui processamento determinístico linear $O(n)$.
- c) AFDs e AFNs são estritamente equivalentes em poder de expressão matemática, pois ambos reconhecem exatamente a mesma classe de Linguagens Regulares (Tipo 3).
- d) Não existe algoritmo matemático capaz de converter um AFN com transições- ϵ em um AFD correspondente.
- e) Todo AFN convertido gera um AFD com o mesmo número de estados do autômato original.

15. (Explosão Combinatória no Pior Caso de Conversão AFN → AFD)

Embora o AFN facilite a abstração e a modelagem gráfica inicial por ser mais compacto, a conversão de um AFN de n estados para um AFD equivalente utilizando a Construção de Subconjuntos pode resultar, no pior caso computacional, em um AFD com um número total de estados de até:

- a) $n + 1$ estados.
- b) $2n$ estados.
- c) n^2 estados.
- d) 2^n estados (crescimento exponencial baseado no Conjunto das Partes).
- e) $n!$ (n fatorial) estados.

16. (Uso do Símbolo de Conjunto Vazio $\emptyset$ na Tabela do AFN)

Ao preencher a Tabela de Transição Matricial de um AFN, a presença do símbolo de conjunto vazio $\emptyset$ em um determinado cruzamento $\delta(q_i, \sigma)$ indica logicamente que:

- a) A máquina aceitou a cadeia automaticamente sem ler o restante da fita.
- b) Aquele ramo específico de computação paralela falhou/morreu por falta de regra de transição para o símbolo lido.
- c) O autômato entrou em estado final de aceitação obrigatório.
- d) O símbolo σ foi convertido automaticamente na palavra vazia ϵ .
- e) O simulador JFLAP encontrou um erro de estouro de memória.

17. (Complexidade Computacional de Execução: Notação $O(n)$)

Em termos de implementação em software e desempenho em tempo de execução para validar uma string de tamanho n , por que a engenharia de compiladores prefere converter modelos AFN em AFDs antes da execução no hardware?

- a) Porque o AFD roda em tempo linear de complexidade $O(n)$, consultando diretamente a matriz de transição a cada símbolo lido, sem necessidade de rastrear múltiplos caminhos ou simular paralelismo.
- b) Porque o AFD exige o dobro de memória RAM para executar o processamento.
- c) Porque o AFN roda com complexidade constante $O(1)$, o que sobrecarrega a CPU.
- d) Porque linguagens de programação reais como Java e C não suportam matrizes.
- e) Porque o AFN é incapaz de processar linguagens binárias.

18. (Interpretando Símbolos Visuais de Autômatos)

Analise os elementos visuais utilizados na convenção gráfica padronizada de autômatos (como no simulador JFLAP):

- I. Um estado representado por um círculo com uma seta entrante isolada indica o Estado Inicial (q_0).
- II. Um estado desenhado com círculos concêntricos (círculo duplo) representa um Estado Final de Aceitação (F).
- III. Setas que saem de um estado e retornam para ele mesmo são denominadas autotransições (self-loops).

É correto o que se afirma em:

- a) I, apenas.
- b) III, apenas.
- c) I e II, apenas.
- d) II e III, apenas.
- e) I, II e III.

19. (Alfabeto da Fita Γ e a Sétupla Formal)

Em exercícios e exames formais, para manter um rigor acadêmico superior e preparar o aluno para modelos avançados, o autômato finito pode ser estendido de uma Quintupla para uma Sétupla $M = (Q, \Sigma, \Gamma, \delta, q_0, F, \sqcup)$. O símbolo especial $\sqcup$ representa:

- a) O estado inicial de emergência do sistema.
- b) O Símbolo Branco, que delimita o final da cadeia preenchida na fita de leitura.
- c) A interrupção do programa por erro léxico.
- d) O operador de fechamento positivo de linguagens.
- e) A interseção dos estados finais com o conjunto das partes.

20. (Aplicação de AFN em Protocolos de Comunicação)

Em redes de computadores, fluxos de dados contínuos chegam em rajadas de bits. Para detectar a chegada de um quadro de dados sincronizado através da busca da subcadeia de pré-âmbulo 1010, a modelagem via AFN simplifica o projeto porque:

- a) Permite manter o estado inicial em um laço ignorando bits irrelevantes e, simultaneamente de forma não-determinística, disparar a verificação da sequência de sincronia no primeiro bit 1 encontrado.
- b) Elimina a necessidade de ter estados finais de aceitação.

- c) Reduz a complexidade da linguagem para Tipo 0 na Hierarquia de Chomsky.
- d) Exige o uso de uma pilha LIFO com marcador de fundo Z_0 .
- e) Substitui a matriz de transição por um algoritmo de ordenação.

PARTE II: 5 Exercícios Práticos para Revisão da N1

Exercício Prático 1: Modelagem Completa de AFD em Tabela e Sétupla Formal

Cenário de Engenharia: Projetar um filtro de validação de dados em hardware que receba um fluxo de bits sobre o alfabeto $\Sigma = \{0, 1\}$ e aceite apenas cadeias que terminem com a sequência rígida 01 (como 001, 10101, 01), rejeitando qualquer outra combinação.

Pede-se:

- A) Apresente a Tabela de Transição de Estados (δ) completa do AFD, incluindo as regras de retorno do determinismo.
- B) Apresente a Sétupla Formal M associada a esta máquina.

Exercício Prático 2: Rastreio de Execução (Teste de Mesa de AFD)

Cenário/Problema: Considerando o AFD projetado no Exercício 1 para reconhecer o sufixo 01, execute o Rastreio Passo a Passo (Teste de Mesa) aplicando a função de transição estendida δ^* para as seguintes cadeias de entrada:

1. Cadeia $w_1 = 1001$
2. Cadeia $w_2 = 0110$

Exercício Prático 3: Bloqueio de Subcadeia Proibida (Estado Morto)

Cenário/Problema: Um firewall de aplicação precisa bloquear qualquer requisição binária que contenha a sequência invasora contígua 11. Se essa sequência aparecer em qualquer ponto da palavra, ela deve ser imediatamente e permanentemente rejeitada.

Objetivo: Construa o AFD sobre $\Sigma = \{0, 1\}$ que aceita apenas palavras que NÃO contêm a subcadeia 11.

Exercício Prático 4: Alternância Rígida de Símbolos

Cenário/Problema: Em um sistema de sinalização de baixo nível sobre o alfabeto $\Sigma = \{a, b\}$, um pacote só é considerado válido se os seus caracteres forem estritamente alternados, ou seja, nunca apresentarem duas letras iguais consecutivas (ex: aceita a, ab, aba, bab, baba; rejeita aab, abb, bba).

Objetivo: Projete a Tabela de Transição do AFD determinístico com estado de erro/lixreira para aceitar a linguagem:

$$L = \{w \mid \text{nenhuma aa ou bb ocorre em } w\}.$$