Teoria da Computação - Teoremas, Lemas e Definições

Renato Avellar Nobre
Ciência da Computação
Universidade de Brasília

Setembro 2018

0 Noções e Terminologia Matemáticas

Alfabeto	Um conjunto finito de objetos chamados símbolos
Cadeia	Uma lista finita de símbolos de um alfabeto
Cadeia vazia	A cadeia de comprimento zero
Complemento	Uma operação sobre um conjunto, formando o conjunto de todos os elementos não presentes
Concatenação	Uma operação que junta cadeias de um conjunto com cadeias de um outro conjunto
Conjunção	Operação booleana E
Conjunto	Um grupo de objetos
Conjunto Vazio	O conjunto sem membros
Elemento	Um objeto em um conjunto
Interseção	Uma operação sobre conjuntos formando o conjunto dos elementos comuns
k-upla	Uma lista de k objetos
Linguagem	Um conjunto de cadeias
Produto Cartesiano	Uma operação sobre conjuntos formando o conjunto de todas as duplas de elementos dos respectivos conjuntos
Símbolo	Um membro de um alfabeto
União	Uma operação sobre conjuntos combinando todos os elementos em um único conjunto

1 Linguagens Regulares

1.1 Autômatos Finitos

Termo - Modelo computacional: Computador idealizado

Definição - 1 Um **autômato finito** é uma 5-upla $(Q, \Sigma, \delta, q_0, F)$, onde

1. Q é um conjunto finito denominado os **estados**,
2. Σ é um conjunto finito denominado o **alfabeto**

3. $\delta : Q \times \Sigma \rightarrow Q$ é a **função de transição**

4. $q_0 \in Q$ é o **estado inicial**, e

5. $F \subseteq Q$ é o **conjunto de estados de aceitação**

Termo - Linguagem da Máquina: Conjunto de todas as cadeias que a máquina aceita $L(M) = A$

Termo - Aceita: Máquinas aceitam cadeias ***Termo - Reconhece:*** Máquinas reconhecem linguagens

Definição - 2 **Definição Formal de Computação para AFD**
Seja $M = (Q, \Sigma, \delta, q_0, F)$ um autômato finito e suponha que $w = w_1, w_2, w_3$, seja uma cadeia onde cada w_i, é um membro do alfabeto Σ. Então M **aceita** w se uma sequencia de estados $r_o, r_1, ..., r_n$ em Q existe com três condições:

1. $r_0 = q_0$

2. $\delta(r_i, w_{i+1}) = r_{i+1}, \text{ para } i = 0, ..., n-1$

3. $r_n \in F$

Definição - 3 Uma linguagem é chamada de uma **linguagem regular** se algum autômato finito a reconhece.

Definição - 4 Sejam A e B linguagens. Definimos as operações regulares união, concatenação, e estrela da seguinte forma.

- **União:** $A \cup B = \{x \mid x \in A \textbf{ ou } x \in B\}$
- **Concatenação:** $A \circ B = \{xy \mid x \in A \textbf{ e } y \in B\}$
- **Estrela:** $A^* = \{x_1, x_2, ..., x_k \mid k \geq 0 \textbf{ e cada } x_i \in A\}$

Termo - Fechada: Uma coleção de objetos é fechada sob alguma operação se, aplicando-se essa operação a membros da coleção, recebe-se um objeto ainda na coleção

Teorema - 1 A classe de linguagens regulares é fechada sob a operação de união

Teorema - 2 A classe de linguagens regulares é fechada sob a operação de concatenação

Teorema - 3 A classe de linguagens regulares é fechada sob a operação de interseção

Teorema - 4 A classe de linguagens regulares é fechada sob a operação estrela

Teorema - 5 A classe de linguagens regulares é fechada sob a operação de complemento

1.2 Não-Determinismo

Definição - 5 Um **autômato finito não-determinístico** é uma 5-upla $(Q, \Sigma, \delta, q_0, F)$, onde

1. Q é um conjunto finito de estados,
2. Σ é um alfabeto finito
3. $\delta : Q \times \Sigma_\epsilon \to P(Q)$ é a **função de transição**
4. $q_0 \in Q$ é o estado inicial, e
5. $F \subseteq Q$ é o conjunto de estados de aceitação

Definição - 6 **Definição Formal de Computação para AFN**
Seja $N = (Q, \Sigma, \delta, q_0, F)$ um autômato finito não determinístico e w uma cadeia sobre o alfabeto Σ. Então dizemos que N **aceita** w se podemos escrever w como $w = y_1 y_2 \ldots y_m$, onde cada y_i é um membro de Σ_ϵ e uma sequencia de estados $r_0, r_1, \ldots, r_m$ existe em Q com três condições:

1. $r_0 = q_0$
2. $r_{i+1} \in \delta(r_i, y_{i+1}),$ para $i = 0, ..., m-1$
3. $r_m \in F$

Teorema - 6 Todo autômato finito não determinístico tem um autômato finito determinístico equivalente

Corolário - 6 Uma linguagem é regular se, e somente se, algum autômato finito não determinístico a reconhecer.

1.3 Expressões Regulares

Definição - 7 Dizemos que R é uma **expressão regular**, se R for

1. a para algum a no alfabeto Σ,
2. ϵ,
3. $\emptyset$,
4. $(R_1 \cup R_2),$ onde R_1 e R_2 são expressões regulares,
5. $(R_1 \circ R_2),$ onde R_1 e R_2 são expressões regulares, ou
6. $(R_1^*),$ onde R_1 é uma expressão regular.

Nos itens 1 e 2, as expressões regulares a e ϵ representam as linguagens $\{a\}$ e $\{\epsilon\}$, respectivamente. No item 3, a expressão regular $\emptyset$ representa a linguagem vazia. Nos itens 4, 5, 6, as expressões representam as linguagens obtidas tomando-se a união ou concatenação das linguagens R_1 e R_2, ou a estrela da linguagem R_1, respectivamente.

Teorema - 7 Uma linguagem é regular se, e somente se, alguma expressão regular a descreve

Lema - 1 Se uma linguagem é descrita por uma expressão regular, então ela é regular

Lema - 2 Se uma linguagem é regular, então ela é descrita por uma expressão regular

Definição - 8 Um **autômato finito não-determinístico generalizado** é uma 5-upla $(Q, \Sigma, \delta, q_0, F)$, onde

1. Q é um conjunto finito de estados,
2. Σ é um alfabeto finito
3. $\delta : (Q - \{q_{fim}\}) \times (Q - \{q_{inicio}\}) \rightarrow \mathcal{R}$ é a **função de transição**
4. $q_{inicio} \in Q$ é o estado inicial, e
5. q_{fim} é o estado de aceitação.

1.4 Linguagens Não Regulares

Lema - 3 **Lema do Bombeamento**

Se A é uma linguagem regular, então há um numero p (o tamanho do bombeamento) onde, se s é qualquer cadeia em A de tamanho de pelo menos p, então s pode ser dividida em três pedações $s = xyz$, satisfazendo as seguintes condições:

1. para cada $i \geq 0, xy^iz \in A$
2. $\mid y \mid > 0$, e
3. $\mid xy \mid \leq p$.

Termo - Princípio da Casa dos Pombos: Se existem p pombos em menos de p casas, algumas casas possuem mais de um pombo. (POR ISSO QUE TEM POMBO NA PORRA TODA)

2 Linguagens Livres-Do-Contexto

2.1 Gramáticas Livres-Do-Contexto

Definição - 9 Uma **gramática livre-do-contexto** é uma 4-upla (V, Σ, R, S), onde:

1. V é um conjunto finito denominado de as **variáveis**,
2. Σ é um conjunto finito, disjunto de V, denominado de os **terminais**,

3. R é um conjunto finito de regras, com cada regra sendo uma variável e uma cadeia de variáveis e terminais, e

4. $s \in V$ é a variável inicial.

Termo - Derivação: Sequencia de substituições para obter uma cadeia

Termo - Árvore Sintática: Representação pictórica de uma derivação

Termo - Linguagem Livre-do-Contexto: Linguagem que pode ser gerada por alguma gramática livre-do-contexto

Termo - Origina: uAv origina uwv, escrito $uAv \Rightarrow uwv$

Termo - Deriva: u deriva v, escrito $u \Rightarrow^* v$, se $u = v$ ou se uma sequência $u_1, u_2, \ldots, u_k$ exite para $k \geq 0$ e $u \Rightarrow u_1 \Rightarrow u_2 \Rightarrow \ldots \Rightarrow u_k \Rightarrow v$

Definição - 10 Uma cadeia w é derivada **ambiguamente** na gramática livre-do-contexto G se ela tem duas ou mais derivações à esquerda diferentes. A gramática G é **ambígua** se ela gera alguma cadeia ambiguamente

Termo - Derivação mais à esquerda: A cada passo da derivação a variável remanescente mais à esquerda é aquela que é substituída.

Termo - Inerentemente Ambíguas: Linguagens livres-do-contexto geradas apenas por gramáticas ambíguas.

Definição - 11 Uma gramática livre-do-contexto está na **forma normal de Chomsky** se toda regra é da forma

$$A \rightarrow BC$$

$$A \rightarrow a$$

onde a é qualquer terminal e A, BeC são quaisquer variáveis, exceto que B e C pode não ser a variável inicial. Adicionalmente, permitimos a regra $S \rightarrow \epsilon$, onde S é a variável inicial.

Termo - Regras unitárias: Da forma $A \Rightarrow B$

Teorema - 8 Qualquer linguagem livre-do-contexto é gerada por uma gramática livre-do-contexto na forma normal de Chomsky

2.2 Autômato com Pilha

Definição - 12 Um **autômato com pilha** é uma 6-upla $(Q, \Sigma, \Gamma, \delta, q_0, F)$, onde Q, Σ, Γ, e F são todos conjuntos finitos, e

1. Q é o conjunto de estados,

2. Σ é o alfabeto de entrada,

3. Γ é o alfabeto de pilha,

4. $\delta : Q \times \Sigma_\epsilon \times \Gamma_\epsilon \to P(Q \times \Gamma_\epsilon)$ é a função de transição,

5. $q_0 \in Q$ é o estado inicial, e

6. $F \subseteq Q$ é o conjunto de estados de aceitação

Definição - 13 **Definição Formal de Computação para AP**
Um autômato com pilha $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ aceita a entrada w se w pode ser escrita como $w = w_1 w_2 \cdots w_m$, onde cada $w_i \in \Sigma_\epsilon$ e sequencias de estados $r_0, r_1, \ldots, r_m \in Q$ e cadeias $s_0, s_1, \ldots, s_m \in \Gamma^*$ existem que satisfazem as três seguintes condições. As cadeias s_i representam a sequencia de conteúdo da pilha que M tem no ramo de aceitação da computação.

1. $r_0 = q_0$ e $s_0 = \epsilon$

2. Para $i = 0, \ldots, m-1$, temos $(r_{i+1}, b) \in \delta(r_i, w_{i+1}, a)$ onde $s_i = at$ e $s_{i+1} = bt$ para algum $a, b \in \Gamma_\epsilon$ e $t \in \Gamma^*$

3. $r_m \in F$

Teorema - 9 Uma linguagem é livre de contexto se, e somente se, algum autômato com pilha a reconhece.

Lema - 4 Se uma linguagem é livre do contexto, então algum autômato com pilha a reconhece.

Lema - 5 Se um autômato com pilha reconhece alguma linguagem, então ela é livre-do-contexto.

2.3 Linguagens Não-Livres-Do-Contexto

Teorema - 10 **Lema do bombeamento para linguagens livres-do-contexto**

Se A é uma linguagem livre-do-contexto, então existe um número p (o comprimento de bombeamento) onde, se s é uma cadeia qualquer em A de comprimento pelo menos p, então s pode ser dividida em cinco partes $s = uvxyz$ satisfazendo as condições:

1. para cada $i \geq 0$, $uv^ixy^iz \in A$

2. $|vy| > 0$, e

3. $|vxy| \leq p$.

3 A Tese de Church-Turing

3.1 Máquinas de Turing

Definição - 14 Uma **máquina de Turing** é uma 7-upla, $(Q, \Sigma, \Gamma, \delta, q_0, q_{aceita}, q_{rejeita})$, onde

Q, Σ, Γ são todos conjuntos finitos e

1. Q é o conjunto de estados,

2. Σ é o alfabeto de entrada não contendo o **símbolo em branco** $\sqcup$,
3. Γ é o alfabeto de fita, onde $\sqcup \in \Gamma$ e $\Sigma \subseteq \Gamma$,
4. $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times E, D$ é a função de transição,
5. $q_0 \in Q$ é o estado inicial,
6. $q_{aceita} \in Q$ é o estado de aceitação, e
7. $q_{rejeita} \in Q$ é o estado de rejeição, onde $q_{rejeita} \neq q_{aceita}$

Termo - Configuração: Um possível valor de estado atual, no conteúdo atual da fita e a proposição atual da cabeça.

Termo - Origina: Uma configuração C_1 origina a configuração C_2, se a máquina de Turing puder ir de C_1 para C_2 em um único passo.

Termo - Configuração Inicial: A máquina está no estado inicial q_0 com sua cabeça na posição mais à esquerda sobre a fita.

Termo - Configurações de Parada: Configuração de aceitação ou rejeição.

Definição - 15 **Definição Formal de Computação para Maquinas de Turing** Uma máquina de Turing M **aceita** a entrada w se uma sequencia de confiigurações $C_1, C_2, \ldots, C_k$ existe, onde

1. C_1 é a configuração inicial de M sobre a entrada w,
2. cada C_i, origina C_{i+1} e
3. C_k é uma configuração de aceitação.

Definição - 16 Chame uma linguagem de **Turing-reconhecível** (ou linguagem recursivamente enumerável) se alguma máquina de Turing a reconhece.

Termo - Decisores: Máquinas de Turing que chegam em configurações de parada sobre a entrada, nunca entrando em loop.

Definição - 17 Chame uma linguagem de Turing-decidível ou simplesmente decidível (ou linguagem recursiva) se alguma máquina de Turing a decide.

3.2 Variantes de Máquinas de Turing

Termo - Robustez: A capacidade da máquina de Turing ter o mesmo poder de suas variantes

Teorema - 11 Toda máquina de Turing multifita tem uma máquina de Turing de uma única fita que lhe é equivalente.

Corolário - 11 Uma linguagem é Turing-reconhecível se, e somente se, alguma máquina de Turing multifita a reconhece.

Teorema - 12 Toda máquina de Turing não-determinística tem uma máquina de Turing determinística que lhe é equivalente.

Corolário - 12 Uma linguagem é Turing-reconhecível se, e somente se, alguma máquina de Turing não-determinística a reconhece.

Termo - Enumerador: Máquina de Turing com impressora em anexo.

Teorema - 13 Uma linguagem é Turing-reconhecível se, e somente se, algum enumerador a enumera.

3.3 A definição do Algoritmo

Termo - Algoritmo: Coleção de instruções simples para realizar una tarefa.

Termo - Tese de Church-Turing: Conexão entre a noção informal de algoritmo e a definição precisa (Noção intuitiva de algoritmos é igual a máquina de Turing). Church usou um sistema notacional denominado de $\lambda - calculo$ para definir algoritmos. Turing o fez com suas "máquinas".

Termo - Descrição Formal: Descrição matemática detalhando todos os aspectos da máquina
Termo - Descrição de implementação: Usamos a linguagem natural para descrever a maneira

pela qual a máquina de Turing move sua cabaça e a forma como ela armazena os dados sobre a fita.
Termo - Descrição de alto-nível: Usamos a linguagem natural para descrever um algoritmo,

ignorando os detalhes de implementação.

4 Decidibilidade

4.1 Linguagens Decidíveis

Teorema - 14 $A_{AFD} = \{\langle B, w\rangle \mid B$ é um AFD que aceita a cadeia de entrada $w\}$ é uma

linguagem decidível.

Teorema - 15 $A_{AFN} = \{\langle B, w\rangle \mid B$ é um AFN que aceita a cadeia de entrada $w\}$ é uma linguagem decidível.

Teorema - 16 $A_{EXR} = \{\langle R, w\rangle \mid R$ é uma expressão regular que gera a cadeia $w\}$ é uma linguagem decidível.

Teorema - 17 $V_{AFD} = \{\langle A\rangle \mid A$ é um AFD e $L(A) = \emptyset\}$ é uma linguagem decidível.

Teorema - 18 $EQ_{AFD} = \{\langle A, B\rangle \mid A$ e B são AFDs e $L(A) = L(B)\}$ é uma linguagem decidível.

Teorema - 19 $A_{GLC} = \{\langle G, w\rangle \mid G$ é uma GLC que gera a cadeia $w\}$ é uma linguagem decidível.

Teorema - 20 $V_{GLC} = \{\langle G\rangle \mid G$ é uma GLC e $L(G) = \emptyset\}$ é uma linguagem decidível.

Teorema - 21 Toda linguagem livre-do-contexto é decidível

4.2 O Problema da Parada

Termo - Método de Diagonalização: Utilizado para a prova de indecidibilidade do problema da parada.

Teorema - 22 $A_{MT} = \{\langle M, w\rangle \mid M$ é uma MT e M aceita$w\}$ é indecidível.

Corolário - 22 Algumas linguagens não são Turing-reconhecíveis.

Definição - 18 Assuma que temos um conjunto A e B e uma função f de A para B. Digamos que f é **um-para-um** se ela nunca mapeia dois elementos diferentes para um mesmo lugar - ou seja, se $f(a) \neq f(b)$ sempre que $a \neq b$. Digamos que f é **sobre** se ela atinge todo elemento de B - ou seja, se para todo $b \in B$ exite um $a \in A$ tal que $f(a) = b$. Digamos que A e B são de **mesmo tamanho** se existe uma função um-para-um e sobre $f : A \to B$. Uma função que é tanto um-para-um quanto sobre é denominada uma **correspondência**. Em uma correspondência todo elemento de A mapeia para um único elemento de B e cada elemento de B tem um único elemento de A mapeando para ele. Uma correspondência é simplesmente uma maneira de emparelhar os elementos de A com os elementos de B.

Definição - 19 Um conjunto A é **contável** se é finito ou tem o mesmo tamanho que $\mathcal{N}$.

Teorema - 23 $\mathcal{R}$ é incontável.

Termo - co-Turing-reconhecível: Linguagem complemento de uma linguagem Turing-reconhecível.

Teorema - 24 Uma linguagem é decidível se ela é Turing-reconhecível e co-Turing-reconhecível, em outras palavras, uma linguagem é decidível exatamente quando ela e seu complemento são ambas Turing-reconhecíveis.

Corolário - 24 $\overline{A_{MT}}$ não é Turing reconhecível.

5 Redutibilidade

Termo - Redutibilidade: Método principal de provar que problemas são computacionalmente insolúveis

Termo - Redução: É uma maneira de converter um problema para um outro de uma forma tal que uma solução para o segundo problema possa ser usada para resolver o primeiro problema. Em termos de teoria da computabilidade, se um problema A é redutível a B e B é decidível, A também é decidível. Equivalentemente, se A é indecidível e redutível a B, B é indecidível.

5.1 Problemas Indecidíveis da Teoria de Linguagens

Teorema - 25 $PARA_{MT} = \{\langle M, w\rangle \mid \text{é uma MT e } M \text{ para sobre a entrada } w\}$ é indecidível

Teorema - 26 $V_{MT} = \{\langle M\rangle \mid M \text{ é uma MT e } L(M) = \emptyset\}$ é indecidível

Teorema - 27 $REGULAR_{MT} = \{\langle M\rangle \mid M \text{ é uma MT e } L(M) \text{ é uma linguagem regular}\}$ é indecidível

Teorema - 28 $EQ_{MT} = \{\langle M_1, M_2\rangle \mid M_1 \text{ e } M_2 \text{ são MTs e } L(M_1) = L(M_2)\}$ é indecidível

Termo - História de Computação: Sequencia de configurações pelas quais a máquina passa à medida que ele processa a entrada. Registro completo da computação dessa máquina.

Definição - 20 Seja M uma máquina de Turing e w uma cadeia de entrada. Uma **historia de computação de aceitação** para M sobre w é uma sequencia de configurações, $C_1, C_2, \ldots, C_l$, onde C_1 é a configuração inicial de M sobre w, C_l é uma configuração de aceitação de M, e cada C_i segue legitimamente de C_{i-1} conforme as regras de M. Uma **história de computação de rejeição** para M sobre w é definida similarmente, exceto que C_l é uma configuração de rejeição.

Definição - 21 Um **autômato linearmente limitado** é um tipo restrito de máquina de Turing na qual a cabeça de leitura-escrita não é permitida mover para fora da parte da fita contendo a entrada. Usando um alfabeto de fita maior que o alfabeto de entrada permite que a memória disponível seja incrementada de no máximo um fator constante.

Teorema - 29 Toda LLC pode ser decidida por um ALL.

Lema - 6 Seja M um ALL com q estados e g símbolos no alfabeto de fita. Existem exatamente qng^n configurações distintas de M para uma fita de comprimento n

Teorema - 30 $A_{ALL} = \{\langle M, w\rangle \mid M \text{ é um ALL que aceita a cadeia } w\}$ é decidível

5.2 Redutibilidade por Mapeamento

Definição - 22 Uma função $f : \Sigma^* \to \Sigma^*$ é uma **função computável** se alguma máquina de Turing M, sobre toda entrada w, para com exatamente $f(w)$ sobre a fita.

Definição - 23 **Definição Formal de Redutibilidade por Mapeamento:** A linguagem A é **redutível por mapeamento** à linguagem B, escrito $A \leq_m B$, se existe uma função computável $f : \Sigma^* \to \Sigma^*$, onde para toda w,

$$w \in A \iff f(w) \in B. \tag{1}$$

A função f é denominada a **redução** de A para B.

Teorema - 31 Se $A \leq_m B$ e B for decidível, então A é decidível.

Corolário - 31 Se $A \leq_m B$ e A for indecidível, então B é indecidível.

Teorema - 32 Se $A \leq_m B$ e B é Turing-Reconhecível, então A é Turing-Reconhecível.

Corolário - 32 Se $A \leq_m B$ e A não é Turing-Reconhecível, então B não é Turing-Reconhecível.

Teorema - 33 EQ_{MT} não é Turing-reconhecível nem co-Turing-reconhecível.

Teorema - 34 **Teorema de Rice** Seja P qualquer propriedade não-trivial de uma linguagem de uma máquina de Turing, onde uma propriedade de funções parciais é chamada trivial se ela vale para todas as funções parciais computáveis ou nenhuma, a linguagem que contém P é não-decidível

6 Tópicos Avançados em Teoria da Computabilidade

6.1 O Teorema da Recursão

Lema - 7 Existe uma função computável $q : \Sigma^* \to \Sigma^*$, onde se w é uma cadeia qualquer, $q(w)$ é a descrição de uma máquina de Turing P_w que imprime w e aí para.

Teorema - 35 **Teorema da recursão:** Seja T uma máquina de Turing que computa uma função $t : \Sigma^* \to \Sigma^*$. Existe uma máquina de Turing R que computa uma função $r : \Sigma^* \to \Sigma^*$, onde para toda w,

$$r(w) = t(\langle R \rangle, w). \tag{2}$$

Definição - 24 Se M é uma máquina de Turing, então dizemos que o *comprimento* da descrição $\langle M \rangle$ de M é o número de símbolos na cadeia descrevendo M. Digamos que M é **mínima** se não existe máquina de Turing equivalente a M que tenha uma descrição mais curta. Seja

$$MIN_{MT} = \{\langle M \rangle \mid M \text{ é uma MT mínima}\} \tag{3}$$

Teorema - 36 MIN_{MT} não é Turing-Reconhecível.

Teorema - 37 Seja $t : \Sigma^* \to \Sigma^*$ uma função computável. Então existe uma máquina de Turing F para a qual $t(\langle F \rangle)$ descreve uma máquina de Turing equivalente a F.

7 Complexidade de Tempo

7.1 Medindo Complexidade

Termo - Análise do Pior-Caso: Tempo de execução mais longo de todas as entradas de um tamanho específico ***Termo - Análise do Caso-Médio:*** Média dos tempos de execução

Definição - 25 Seja M uma máquina de Turing determinística que para sobre todas as entradas. O **tempo de execução** ou **complexidade de tempo** de M é a função $f : \mathcal{N} \to \mathcal{N}$, onde $f(n)$ é o número máximo de passos que M usa sobre qualquer entrada de comprimento n. Se $f(n)$ for o tempo de execução de M, dizemos que M roda em tempo $f(n)$ e que M é uma maquina de Turing de tempo $f(n)$. Costumeiramente usamos n para representar o comprimento de entrada.

Termo - Análise Assintótica: Entender o tempo de execução sobre entradas grandes, considerando apenas o termo de mais alta ordem da expressão par o tempo de execução do algoritmo.

Definição - 26 **Notação O-grande:** Sejam f e g funções $f, g : \mathcal{N} \to \mathcal{R}^+$. Vamos dizer que $f(n) = O(g(n))$ se inteiros positivos c e n_0 existem tais que para todo inteiro $n \geq n_0$

$$f(n) \leq cg(n).$$

Quando $f(n) = O(g(n))$ dizemos que $g(n)$ é um **limitante superior assintótico** para $f(n)$.

Termo - Limitantes polinomiais: Limitantes da forma n^c para c maior que 0. ***Termo - Limitantes exponenciais:*** Limitantes da forma $2^{(n^\delta)}$ com $\delta > 0$

Definição - 27 **Notação o-pequeno:** Sejam f e g funções $f, g : \mathcal{N} \to \mathcal{R}^+$. Dizemos que $f(n) = o(g(n))$ se

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$$

Em outras palavras, $f(n) = o(g(n))$ significa que, para qualquer número real $c > 0$, um número n_0 existe, onde $f(n) < cg(n)$ para todo $n \geq n_0$.

Definição - 28 Seja $t : \mathcal{N} \to \mathcal{R}^+$ uma função. Defina a **classe de complexidade de tempo** $TIME(t(n))$, como sendo a coleção de todas as linguagens que são decidíveis por uma máquina de Turing de tempo $O(t(n))$

Termo - Tempo linear: Tempo $O(n)$

Teorema - 38 Seja $t(n)$ uma função, onde $t(n) \geq n$. Então toda máquina de Turing multifita de tempo $t(n)$ tem uma máquina de Turing de uma única fita equivalente de tempo $O(t^2(n))$.

Definição - 29 Seja $\mathcal{N}$ uma máquina de Turing não-deterministíca que é um decisor. O **tempo de execução** de $\mathcal{N}$ é uma função $f : \mathcal{N} \to \mathcal{N}$, onde $f(n)$ é o número máximo de passos que $\mathcal{N}$ usa sobre qualquer ramo de sua computação sobre qualquer entrada de comprimento n

Teorema - 39 Seja $t(n)$ uma função, onde $t(n) \geq n$. Então toda máquina de Turing não-determinística de uma-única-fita de tempo $t(n)$ tem uma máquina de Turing determinística de uma-única-fita de tempo $2^{O(t(n))}$

7.2 A Classe P

Definição - 30 **P** é a classe de linguagens que são decidíveis em tempo polinomial sobre uma máquina de Turing determinística de uma-única-fita. Em outras palavras,

$$P = \bigcup_k TIME(n^k)$$

Teorema - 40 $CAMINH = \{\langle G, s, t\rangle \mid G$ é um grafo direcionado que tem um caminho de s para $t\} \in P$

Teorema - 41 $RELPRIME = \{\langle x, y\rangle \mid x$ e y são primos entre si$\} \in P$

Termo - Algoritmo euclidiano: Algoritmo usado para calcular o máximo divisor comum de números naturais em tempo polinomial.

Teorema - 42 Toda linguagem livre-do-contexto é um membro de P.

7.3 A Classe NP

Definição - 31 Um **verificador** para uma linguagem A é um algoritmo V, onde

$$A = \{w \mid V \text{ aceita } \langle w, c\rangle \text{ para alguma cadeia } c\}$$

Medimos o tempo de um verificador somente em termos do comprimento de w, portanto u **verificador de tempo polinomial** roda em tempo polinomial no comprimento de w. Uma linguagem A é **polinomialmente verificável** se ela tem um verificador de tempo polinomial.

Definição - 32 **NP** é a classe de linguagens que têm verificadores de tempo polinomial

Teorema - 43 Uma linguagem está em NP se, e somente se, ela é decidida por alguma máquina de Turing, não-determinística de tempo polinomial.

Definição - 33 **NTIME*(t(n))*** $= \{L \mid L$ é uma linguagem decidida por uma máquina de Turing não-determinística de tempo $O(t(n))\}$.

Corolário - 43 NP $= \bigcup_k NTIME(n^k)$.

Teorema - 44 $CLIQUE = \{\langle G, k\rangle \mid G$ é um grafo não direcionado com um k-clique$\}$ está em NP

Teorema - 45 $SUBSETSUM = \{\langle S, t\rangle \mid S = \{x_1, \ldots, x_k\}$ e para algum $\{y_1, \ldots, y_t\} \subseteq \{x_1, \ldots, x_k\}$, temos $\Sigma y_i = t\}$ está em NP.

7.4 NP-Completude

Termo - NP-completos: Problemas em NP cuja complexidade individual está relacionada aquela da classe inteira.

Termo - Problema da Satisfatibilidade: Problema NP-completo que consiste em dizer se uma fórmula booleana é **satisfativel**, ou seja, se alguma atribuição de 0s e 1s às variáveis faz a formula ter valor 1.

Teorema - 46 **Teorema de Cook-Levin**
$SAT = \{\langle\phi\rangle \mid \phi$ é uma fórmula boolean satisfatível $\} \in$ P se, e somente se, P = NP.

Definição - 34 Uma função $f : \Sigma^* \rightarrow \Sigma^*$ é uma **função computável em tempo polinomial** se alguma máquina de Turing de tempo polinomial M existe que para com exatamente $f(w)$ na sua fita, quando iniciada sobre qualquer entrada w.

Definição - 35 A linguagem A é **redutível por mapeamento em tempo polinomial** à linguagem B, escrito $A \leq_P B$, se existe uma função computável $f : \Sigma^* \rightarrow \Sigma^*$, onde para toda w,

$$w \in A \iff f(w) \in B. \tag{4}$$

A função f é denominada a **redução em tempo polinomial** de A para B.

Teorema - 47 Se $A \leq_P B$ e $B \in P$, então $A \in P$.

Termo - Literal: Variável booleana ou sua negação

Termo - Cláusula: Fórmula composta de vários literais conectados por $\vee$

Termo - Forma normal conjuntiva ou fnc-fórmula: Cláusulas conectadas por $\wedge$

Termo - 3fnc-fórmula: fnc-fórmula em que todas as cláusula existem exatamente 3 literais

Teorema - 48 $3SAT = \{\langle\phi\rangle \mid \phi$ é uma 3fnc-fórmula satisfatível$\}$ é redutível em tempo polinomial a $CLIQUE$

Definição - 36 Uma liinguagem B é *NP-completa* se ela satisfaz duas condições:

1. B está em NP, e
2. toda A em NP é redutível em tempo polinomial a B.

Teorema - 49 Se B for NP-completa e $B \in P$, então P = NP

Teorema - 50 Se B for NP-completa e $B \leq_P C$ para $C \in$ NP, então C é NP-completa

Teorema - 51 SAT é NP-completo. Esse teorema re-enuncia o Teorema de Cook-Levin

Corolário - 51 $3SAT$ é NP-completa.