

Disciplinas MCTA015-13 / CCM-104

Linguagens Formais e Autômatas / Teoria da Computação

Profa. Carla Negri Lintzmayer

`carla.negri@ufabc.edu.br`

`www.professor.ufabc.edu.br/~carla.negri`

Centro de Matemática, Computação e Cognição – Universidade Federal do ABC



Estes slides não contêm um conteúdo completo sobre MTs: são apenas um apoio gráfico a uma aula específica dada em sala.

Cadeias e MTs

Cadeias podem representar qualquer objeto:

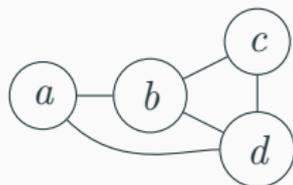
- **Números:** 5 \rightarrow “5” ou “00110101” ou “101”, etc ...

Cadeias podem representar qualquer objeto:

- **Números:** $5 \rightarrow$ "5" ou "00110101" ou "101", etc ...
- **Polinômios:** $4x^2 + 3y - 5 \rightarrow$ "4x^2 + 3y - 5" ou "3,2,4x2+3y1-5y0" ...

Cadeias podem representar qualquer objeto:

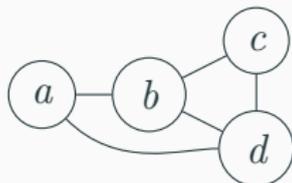
- **Números:** $5 \rightarrow$ "5" ou "00110101" ou "101", etc ...
- **Polinômios:** $4x^2 + 3y - 5 \rightarrow$ "4x² + 3y - 5" ou "3,2,4x²+3y1-5y⁰" ...
- **Grafos:**



\rightarrow "({a,b,c,d},{ab,bc,cd,bd,ad})"

Cadeias podem representar qualquer objeto:

- **Números:** $5 \rightarrow "5"$ ou $"00110101"$ ou $"101"$, etc ...
- **Polinômios:** $4x^2 + 3y - 5 \rightarrow "4x^2 + 3y - 5"$ ou $"3,2,4x^2+3y1-5y0"$...
- **Grafos:**



$\rightarrow "(\{a, b, c, d\}, \{ab, bc, cd, bd, ad\})"$

- **Imagens:**



187	182	174	148	162	162	167	171	141	158	164	
186	142	133	74	75	62	59	57	270	140	154	
186	142	82	14	44	4	12	24	88	140	141	
204	154	4	124	131	111	104	104	75	54	140	
194	86	138	287	287	288	228	227	87	7	201	
173	105	207	238	214	220	238	238	74	74	204	
186	86	178	289	180	218	211	188	138	74	201	
148	52	144	44	88	134	134	51	51	62	22	148
189	148	141	192	192	227	174	143	102	114	26	194
205	174	148	292	294	297	249	178	224	83	224	204
196	214	174	148	234	187	88	148	74	28	224	241
194	234	147	148	227	218	227	121	94	21	205	204
194	214	173	86	132	143	34	84	7	28	149	218
187	186	224	74	1	47	4	6	117	205	231	
187	202	227	148	4	12	108	249	148	149	204	
194	204	123	207	177	121	123	209	174	12	108	218

187	182	174	148	162	162	167	171	141	158	164	
186	142	133	74	75	62	59	57	270	140	154	
204	154	4	124	131	111	104	104	75	54	140	
194	86	138	287	287	288	228	227	87	7	201	
173	105	207	238	214	220	238	238	74	74	204	
186	86	178	289	180	218	211	188	138	74	201	
148	52	144	44	88	134	134	51	51	62	22	148
189	148	141	192	192	227	174	143	102	114	26	194
205	174	148	292	294	297	249	178	224	83	224	204
196	214	174	148	234	187	88	148	74	28	224	241
194	234	147	148	227	218	227	121	94	21	205	204
194	214	173	86	132	143	34	84	7	28	149	218
187	186	224	74	1	47	4	6	117	205	231	
187	202	227	148	4	12	108	249	148	149	204	
194	204	123	207	177	121	123	209	174	12	108	218

Cadeias podem representar qualquer objeto:

- AFDs:



“ $((q1, q2), (a, b), ((q1, a, q2), (q1, b, q2), (q2, a, q2), (q2, b, q2)), q1, (q2))$ ”

- Máquinas de Turing recebem *cadeias*.
 - Então, qualquer outro objeto deve ser convertido primeiro.
 - A máquina pode ser “programada” para decodificar a representação e interpretá-la como queremos.

- Máquinas de Turing recebem *cadeias*.
 - Então, qualquer outro objeto deve ser convertido primeiro.
 - A máquina pode ser “programada” para decodificar a representação e interpretá-la como queremos.
- Descrições de alto nível não detalham a codificação.
 - Se a entrada for ω , assumiremos que é uma cadeia qualquer.

- Máquinas de Turing recebem *cadeias*.
 - Então, qualquer outro objeto deve ser convertido primeiro.
 - A máquina pode ser “programada” para decodificar a representação e interpretá-la como queremos.
- Descrições de alto nível não detalham a codificação.
 - Se a entrada for ω , assumiremos que é uma cadeia qualquer.
 - Se for $\langle A \rangle$, então será um objeto A codificado em cadeia.

- Máquinas de Turing recebem *cadeias*.
 - Então, qualquer outro objeto deve ser convertido primeiro.
 - A máquina pode ser “programada” para decodificar a representação e interpretá-la como queremos.
- Descrições de alto nível não detalham a codificação.
 - Se a entrada for ω , assumiremos que é uma cadeia qualquer.
 - Se for $\langle A \rangle$, então será um objeto A codificado em cadeia. Assumiremos que a máquina implicitamente testa se a codificação está apropriada.

Exemplo

Considere a linguagem $E = \{\langle G \rangle : G \text{ é um grafo conexo}\}$.

Considere a linguagem $E = \{\langle G \rangle : G \text{ é um grafo conexo}\}$.

$M_E =$ "Sobre a entrada $\langle G \rangle$, onde G é um grafo:

1. Selecione um vértice de G e marque-o.
2. Repita até que nenhum novo vértice seja marcado:
 - 2.1 Marque um vértice que tenha aresta para um vértice já marcado.
3. Se todos os vértices foram marcados, aceite. Caso contrário, rejeite."