IMPLEMENTAÇÃO DO ALGORITMO DE ORDENAÇÃO BUBBLE SORT UTILIZANDO MÁQUINA DE TURING

Disciplina: Linguagens Formais e Autômatos

Orientador: Dr. Thales L. A. Valente



Universidade Federal do Maranhão

São Luís - Maranhão. | 11 e 13 de fev. 2025

INTEGRANTES - GRUPO 3

FERNANDA SOUSA DE ASSUNÇÃO VALE

Matrícula: 2022024316

JHONES DE SOUSA SOARES

Matrícula: 2020002730

ÍNDICE

04	Introdução
09	Fundamentação Teórica
12	Desenvolvimento
21	Análise de Resultados
28	Conclusão
29	Programa em Funcionamento
30	Referências

- O que é a Máquina de Turing?
- Modelo teórico criado por Alan Turing (1936);
- Define quais problemas podem ser resolvidos por um algoritmo;
- Usada para estudar as possibilidades e os limites da computação

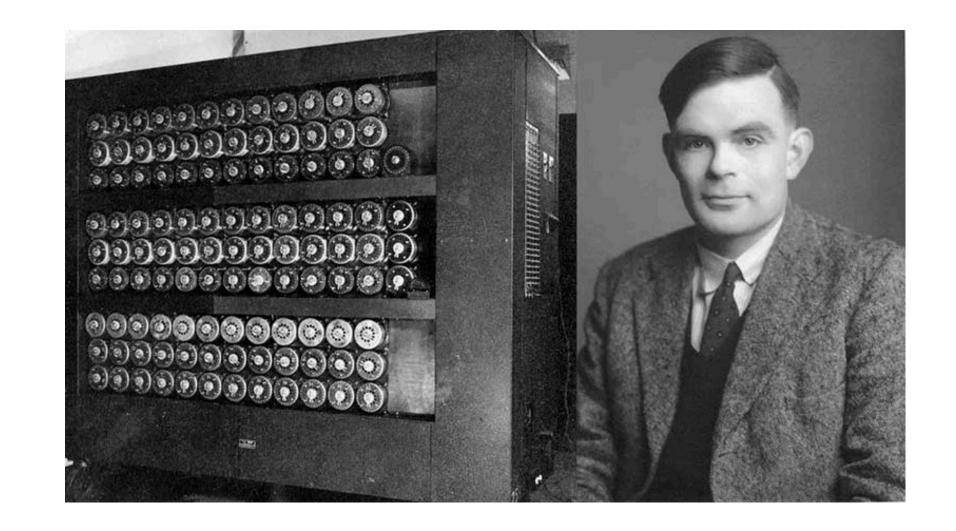


Figura 1.1 - Alan Turing

Componentes principais:

- Fita infinita (memória de entrada e saída)
 à direita e dividida em "casas":
- Cabeçote de leitura/escrita, estando sempre situado sobre uma das casas da fita;
- Conjunto de estados e tabela de transição.

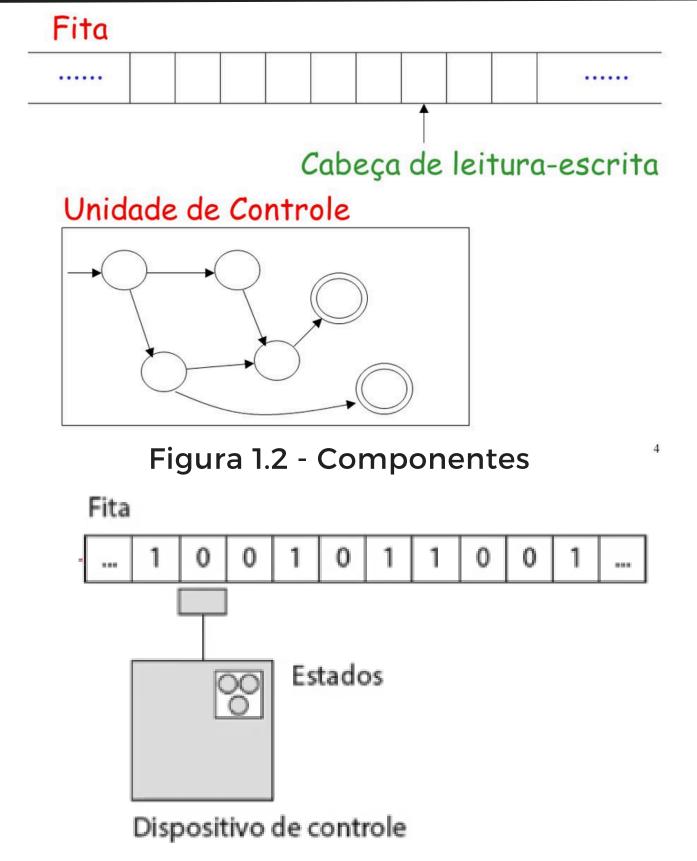
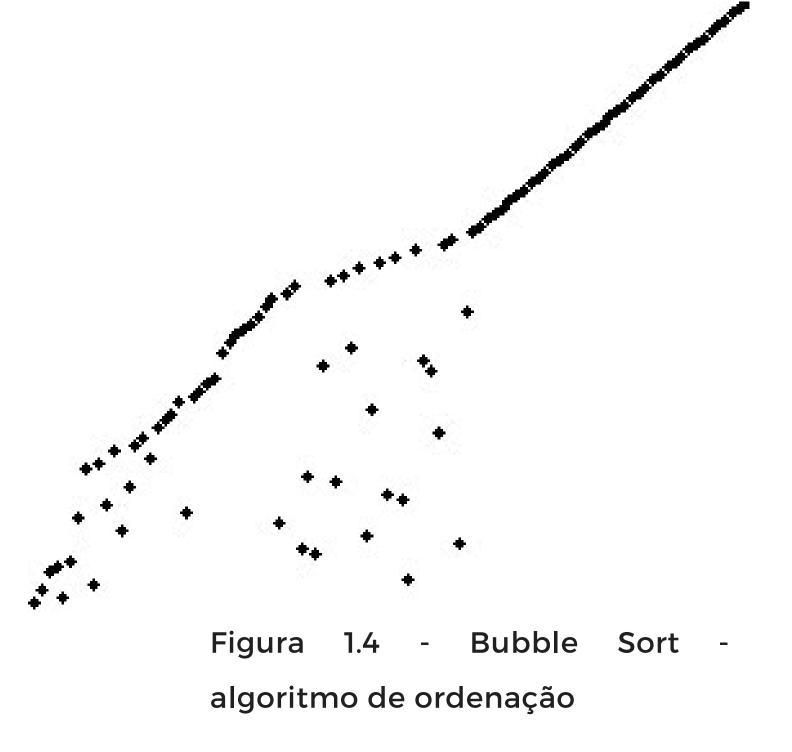


Figura 1.3 - Outro exemplo de componentes

O que é o Bubble Sort?

- Algoritmo de ordenação simples, mas ineficiente na ordenação de um grande conjunto;
- Compara elementos adjacentes e os troca caso estejam fora de ordem.
- Repetido até que a lista esteja ordenada.



Objetivos do trabalho

- Apresentar a Máquina de Turing e seu papel na computação.
- Explicar o funcionamento do Bubble Sort e sua relação com a Máquina de Turing.

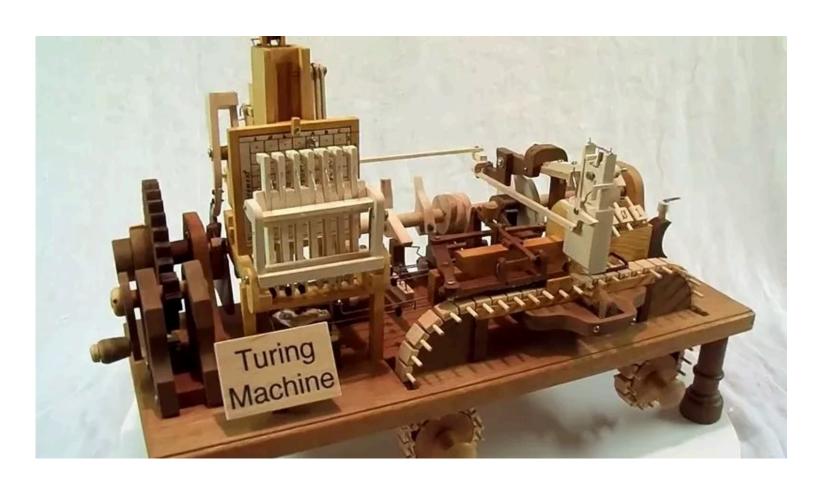


Figura 1.5 - Máquina de Turing

Bubble Sort

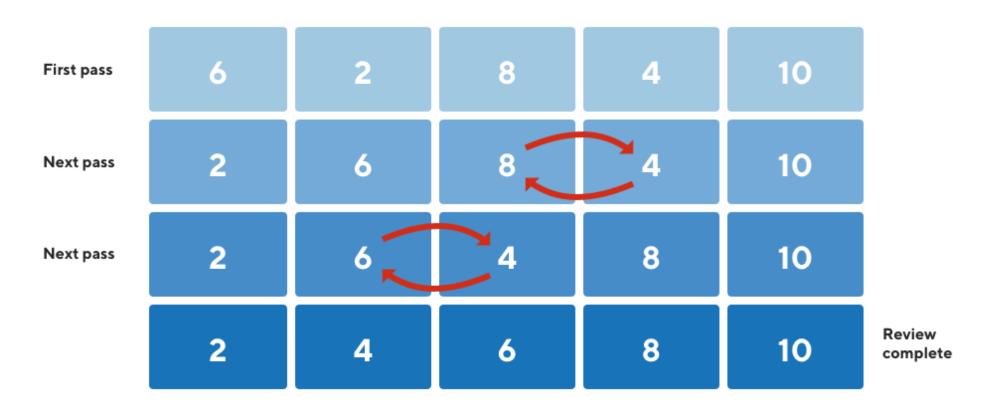


Figura 1.6 - Representação do Bubble Sort

Relação entre a Máquina de Turing e o Bubble Sort

O Bubble Sort funciona comparando elementos adjacentes e trocando-os se estiverem fora de ordem. A Máquina de Turing pode implementar essa lógica percorrendo a fita, lendo valores e reescrevendo conforme necessário.

2. FUNDAMENTAÇÃO TEÓRICA

Máquina de Turing - Definição formal:

Máquina de Turing com Fita Infinita M é uma 8-tupla:

$$M = \{\Sigma, Q, \delta, q_0, F, V, \beta, \circledast\}$$

- Σ: alfabeto de símbolos de entrada;
- Q: conjunto finito de estados do autômato;
- δ: função programa da forma

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

- q0: estado inicial, tal que q0 \in Q.
- F: conjunto de estados finais, tal que $F \subseteq Q$.
- V: alfabeto auxiliar (pode ser vazio).
- β: símbolo especial branco.
- 🟵: símbolo ou marcador de início da fita.

2. FUNDAMENTAÇÃO TEÓRICA

Máquina de Turing - FUNCIONAMENTO:

Possui um controle de estados finitos e uma fita de leitura/escrita, onde:

- 1.O cabeçote lê um símbolo na fita.
- 2. Seguindo a tabela de transição, ele:
 - a. Escreve um novo símbolo.
 - b. Move-se para a esquerda ou direita.
 - c. Muda de estado.
- 3. Repete até atingir um estado de parada

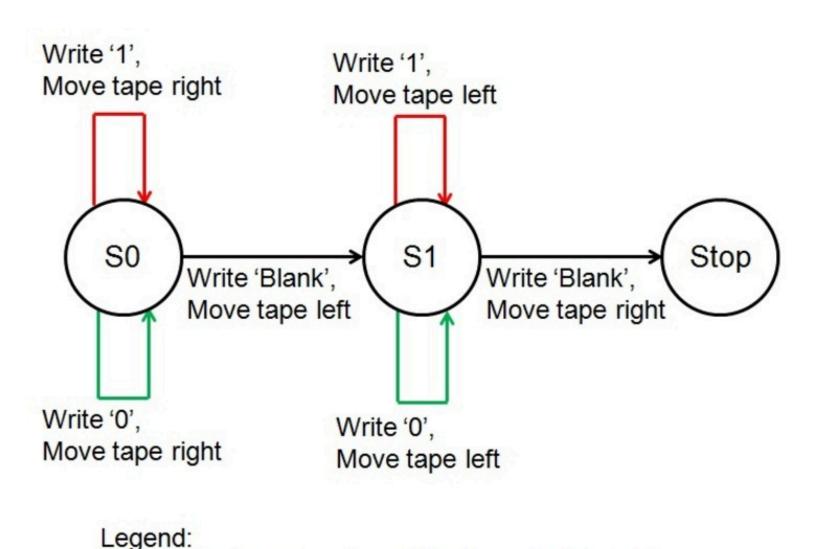


Figura 2.1 - Instruções de uma Máquina de Turing

Instruction when '0' symbol is read

Instruction when '1' symbol is read

Instruction when 'Blank' symbol is read

2. FUNDAMENTAÇÃO TEÓRICA

- Bubble Sort FUNCIONAMENTO:
 - 1. Comparar dois elementos adjacentes: se o primeiro é maior do que o segundo, ambos são trocados;
 - 2. Realizar a comparação/troca (caso necessária) até o fim da lista. Neste ponto o último elemento é o maior.
 - 3. Repetir o passo 2 para todos os elementos até que nenhuma troca seja necessária, com exceção do último sucessivamente.

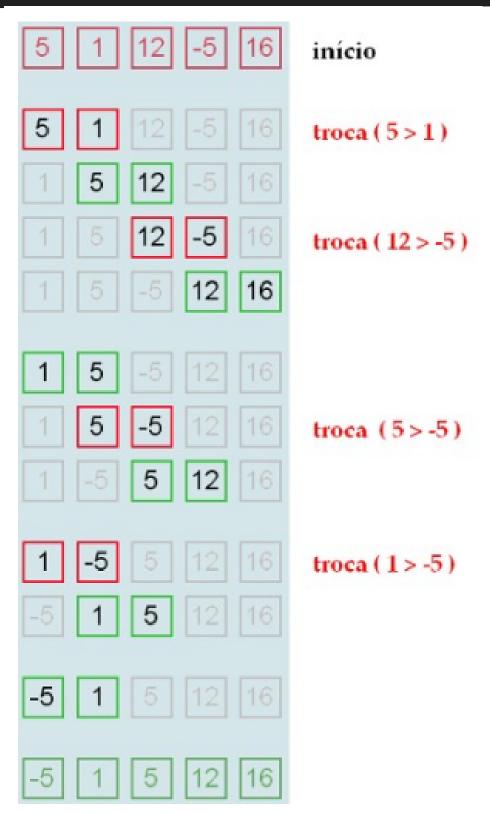


Figura 2.2 - Demonstração do Bubble Sort

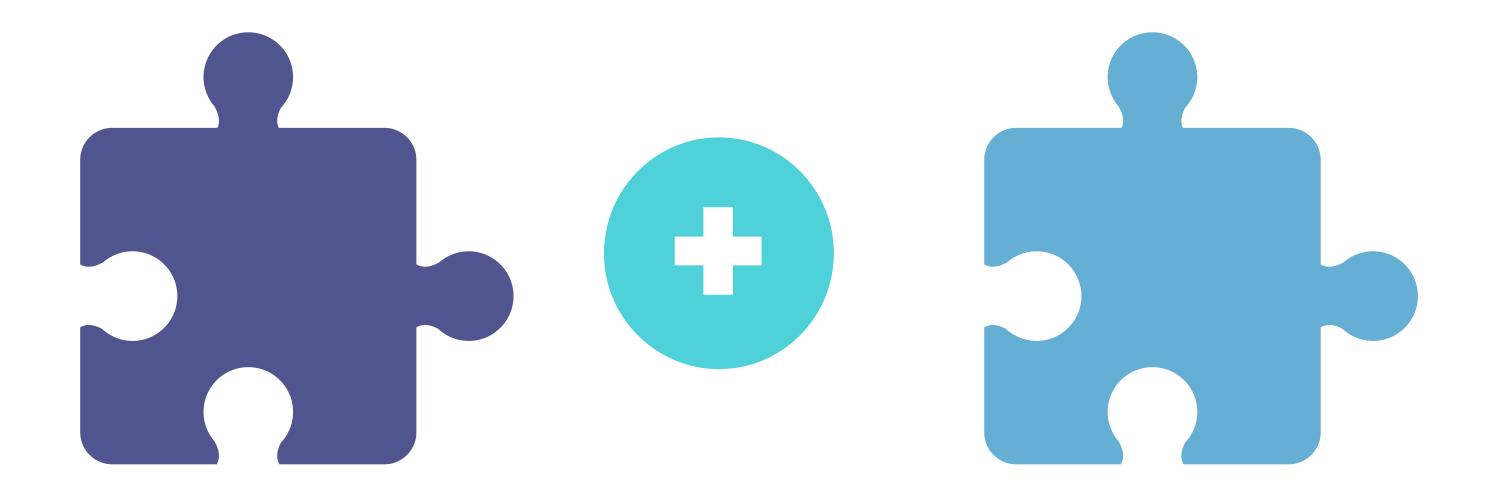
3. DESENVOLVIMENTO

PARTE 1

Codigo "turing-machine.ipynb".

PARTE 2

Visualização da fita



Definição formal do algoritmo

```
TM = (Q, \Sigma, \Gamma, \delta, q_0, q_accept)
Q (Estados): {Inicio(q0), comparação, troca, checar_flag, fim(q_accept)}
\Sigma (Alfabeto de entrada): {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
\Gamma (Alfabeto da fita): \Sigma \cup \{0, 1, \square\} (Indicadores de troca)
\delta (Funções de transição): definido na tabela de transição
q_0 (Estado inicial): Inicio
q_accept: Fim
```

Tabela de transição

Estado Atual	Simbolo Lido	Próximo estado	Simbolo escrito	Direção do movimento	Descrição
Inicio(q0)	*	Comparação(q1)	Comparação(q1) * Direta		Começo de um novo ciclo
Comparação(q1)	а	Comparação(q1)	1) a Direta		a ≤ próximo elemento
Comparação(q1)	а	Troca(q2)	а	-	a > próximo elemento
Troca(q2)	а	Comparação(q1)	b	Direta	Troca a↔b, alteração da flag
Comparação(q1)		Checar_flag(q3)		Esquerda	Fim da lista
Checar_flag(q3)	0	Fim(q_accepted)	epted) O -		Sem trocas → Ordenado
Checar_flag(q3)	1	Inicio(q0)	0	Esquerda Resetar fla cicl	

Tabela 3.1 - Tabela de estados de transição da maquina de turing com algoritmo de

ordenação bubble sort

Diagrama

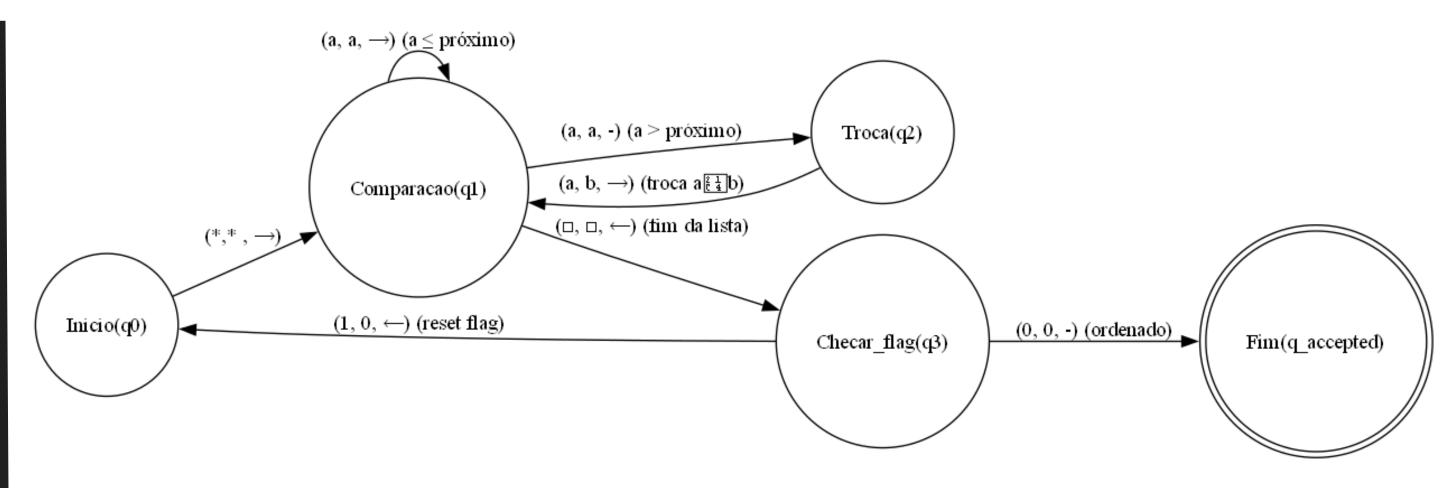


Figura 3.1 - Diagrama de estados de transição para a maquina de turing

Exemplo de uso tabela de transição

Ordenação inicial da

lista: 3,1,2,4

Passo	Estado	Fita(posição da cabeça)	Ação	
1	Inicio(q0)	[3,1,2,4,0]	Reseta cabeça de leitura	
2	Comparação(q1)	[3 <u>,</u> 1,2,4,0]	Compara 3 > 1 → Troca	
3	Troca(q2)	[1 <u>,</u> 3,2,4,1]	Troca completa	
4	4 Comparação(q1) [1,3 <u>,</u> 2,4,1]		Compara 3 > 2 → Troca	
5	Troca(q2)	[1,2 <u>,</u> 3,4,1]	Troca completa	
6	Comparação(q1)	[1,2,3 <u>,</u> 4,1]	Compara 3 < 4 → Move	
7	Checar_flag(q3)	[1,2,3,4,1]	Flag=1 → Novo ciclo	
8	8 Inicio(q0) [1 <u>,</u> 2,3,4,0]		Reseta para o novo ciclo	
9	9 Comparação(q1)		Todos os elementos ordenados → fim	

Tabela 3.2 - Tabela com estados de transição para um exemplo teste do algoritmo de ordenação

PARTE 1 Código turingmachine.ipynb

• Implementação do Bubble Sort

- Estado 'inicio': Reinicia a fita, posiciona a cabeça na primeira posição;
- Estado 'comparar': Compara elemento atual com o próximo. Se maior, troca;
- Estado 'trocar': Realiza a troca e move a cabeça para a próxima posição;
- Estado 'verificar_flag': Verifica se houve troca. Se sim, reinicia. Caso contrário, a ordenação está concluída;

PARTE 1 Código turingmachine.ipynb

Máquina de Turing

- Fita: Representa os elementos a serem ordenados;
- Cabeça de leitura: Realiza comparações e troca de valores;
- Fluxo: A cada passo, a máquina verifica e troca valores, simulando o Bubble Sort.

PARTE 2 Visualização

Fluxo de Execução

- Fita inicial: ['1', '4', '4', '3', '2'];
- A máquina executa até a ordenação completa;
- A animação mostra o progresso a cada passo.

PARTE 2 Visualização

Animação

- Biblioteca: Usamos matplotlib.animation para criar a animação;
- Quadros: Cada quadro representa um passo do algoritmo;
- Informações Exibidas: Fita, estado da máquina, flag de troca, e log de ações.

4. ANÁLISE DE RESULTADOS

Estado Atual	Simbolo lido	Próximo estado	Simbolo escrito	Direção do movimento	Condição
q0	3	q1	3	Direita	Movimenta para o próximo
q1	2	q2	2	Esquerda	Troca necessária
q2	3	q2	2	Direita	Troca para a esquerda
q2	2	q1	3	Direta	Troca para a direita
q1	1	q2	1	Esquerda	Troca necessária
q2	3	q2	1	Direita	Troca para a esquerda
q2	1	q1	3	Direita	Troca para a Direta
q1	_	q3	_	Esquerda	Fim da fita, checar flag de troca
q3	X	q0	X	Esquerda	Reiniciar se houver alguma troca
q3	_	q4		_	Fim da ordenação

Tabela 4.1 - Tabela de transições para ordenação da lista (3,2,1)

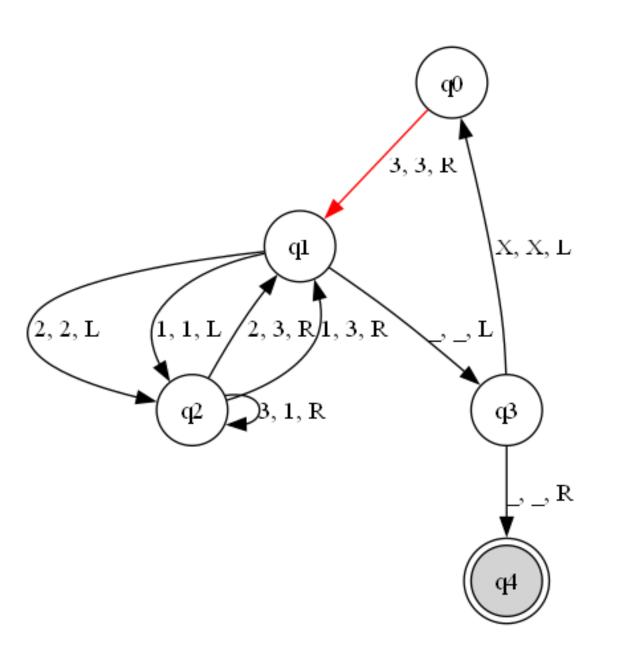


Figura 4.1 - Diagrama de estados da ordenação obtida - Passo 1

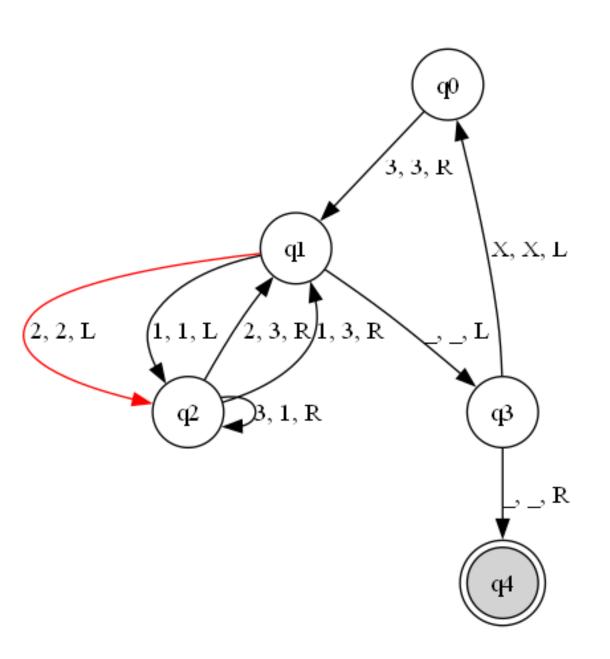


Figura 4.1 - Diagrama de estados da ordenação obtida - Passo 2

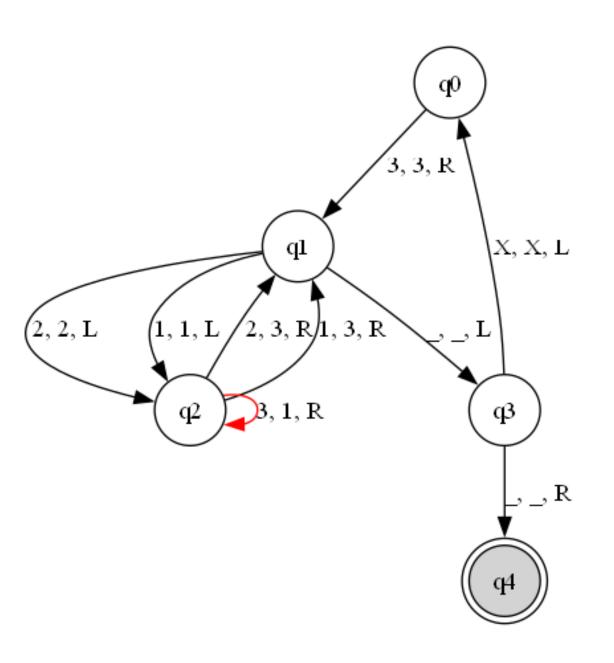


Figura 4.1 - Diagrama de estados da ordenação obtida - Passo 3

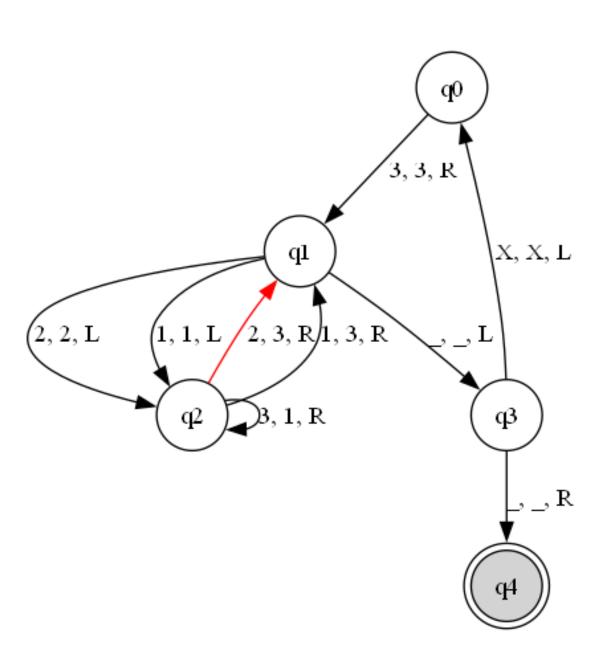


Figura 4.1 - Diagrama de estados da ordenação obtida - Passo 4

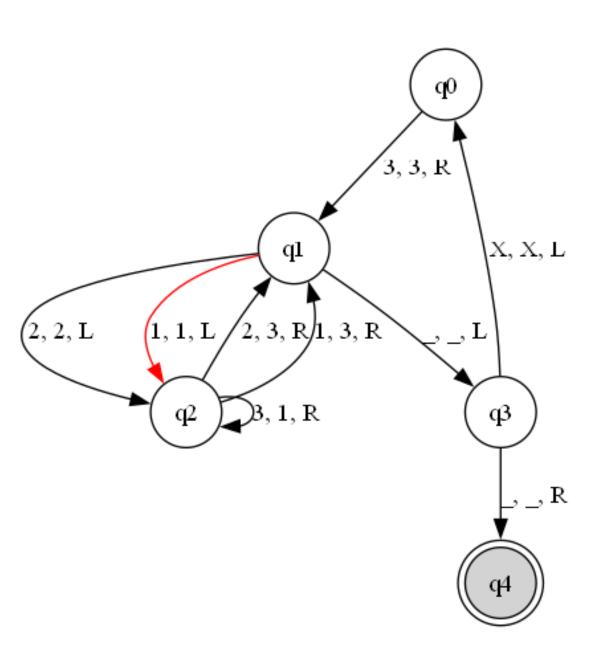


Figura 4.1 - Diagrama de estados da ordenação obtida - Passo 5

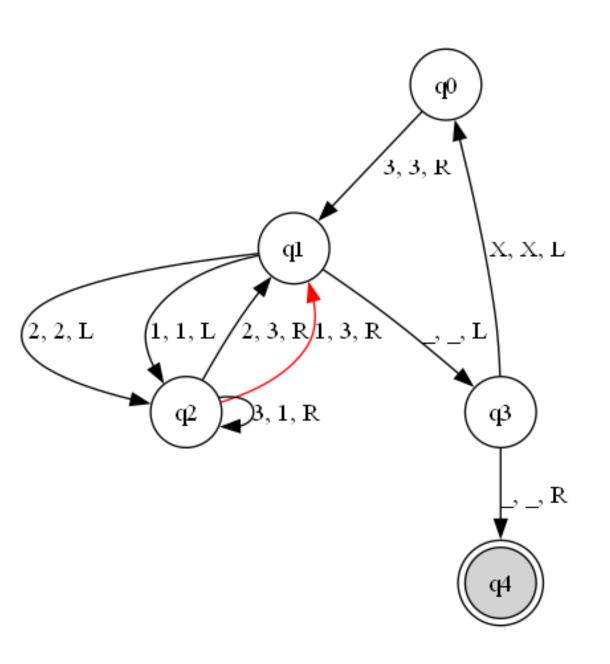


Figura 4.1 - Diagrama de estados da ordenação obtida - Passo 6

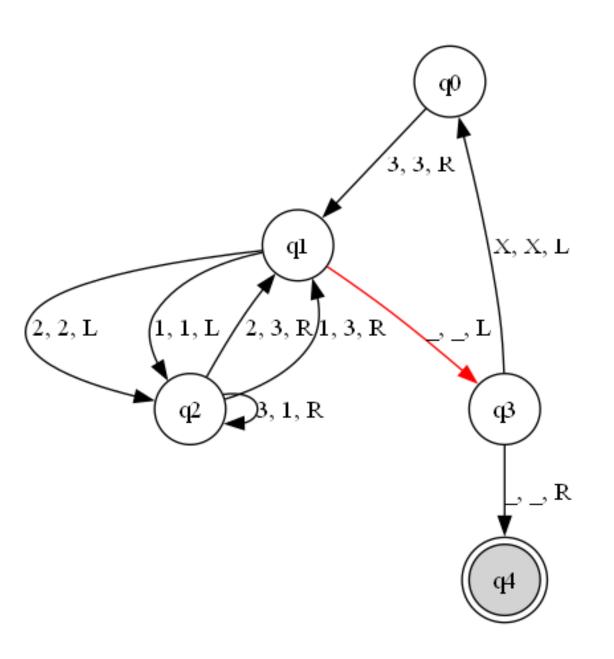


Figura 4.1 - Diagrama de estados da ordenação obtida - Passo 7

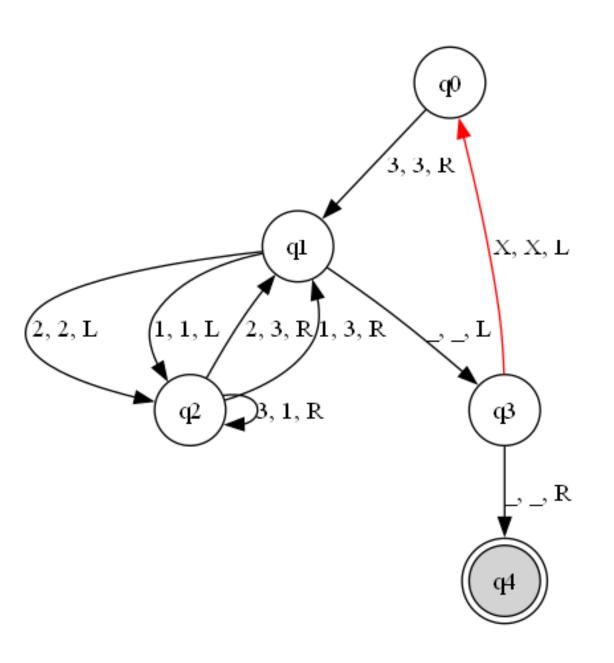


Figura 4.1 - Diagrama de estados da ordenação obtida - Passo 8

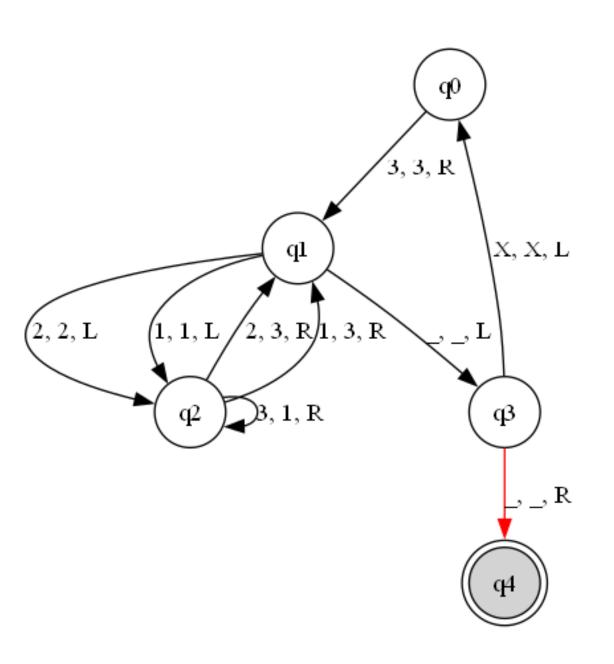


Figura 4.1 - Diagrama de estados da ordenação obtida - Passo 9

Estado Atual	Simbolo lido	Próximo estado	Simbolo escrito	Direção do movimento	Condição
q0	3	q1	3	Direita	Movimenta para o próximo
q1	2	q2	2	Esquerda	Troca necessária
q2	3	q2	2	Direita	Troca para a esquerda
q2	2	q1	3	Direta	Troca para a direita
q1	1	q2	1	Esquerda	Troca necessária
q2	3	q2	1	Direita	Troca para a esquerda
q2	1	q1	3	Direita	Troca para a Direta
q2	4	q1	4	Direita	Sem trocas
q1	_	q3	_	Esquerda	Fim da fita, checar flag de troca
q3	Χ	q0	X	Esquerda	Reiniciar se houver alguma troca
q3		q4		_	Fim da ordenação

Tabela 4.2 - Tabela de transições para ordenação da lista (3,2,1,4)

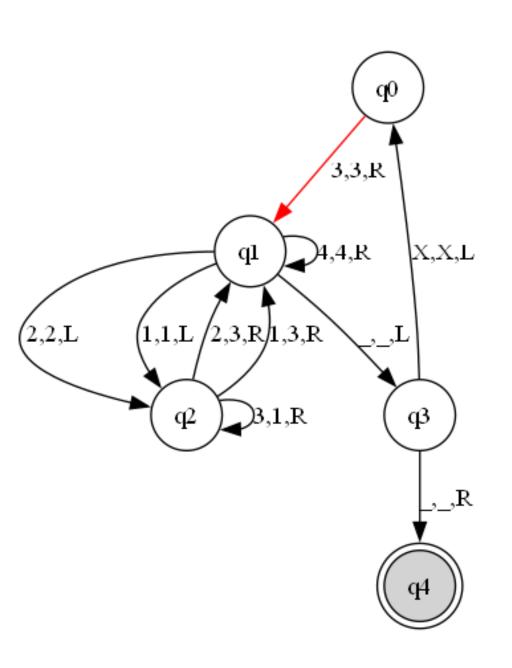


Figura 4.2 - Diagrama de estados da ordenação obtida - Passo 1

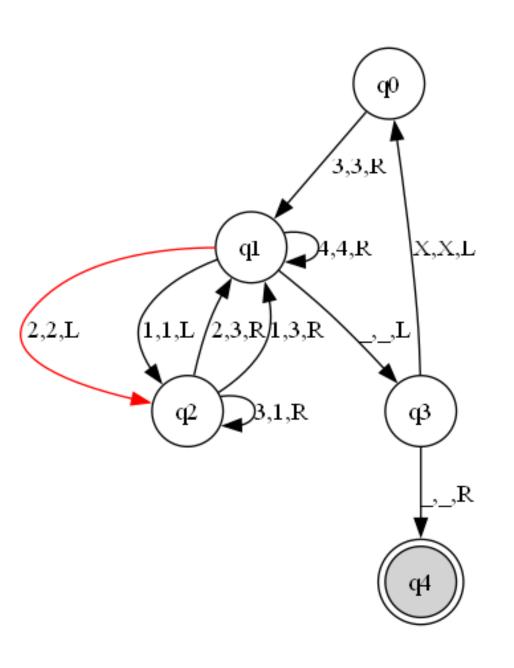


Figura 4.2 - Diagrama de estados da ordenação obtida - Passo 2

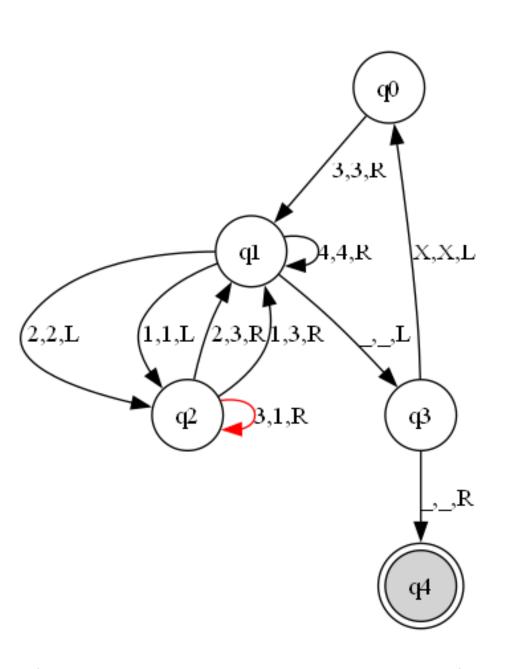


Figura 4.2 - Diagrama de estados da ordenação obtida - Passo 3

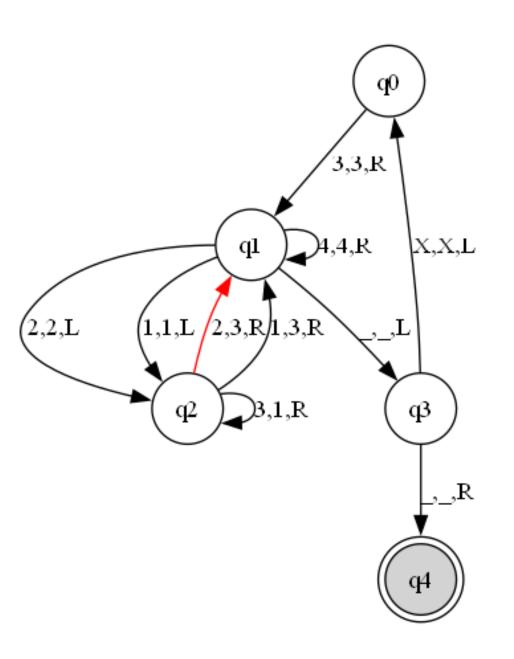


Figura 4.2 - Diagrama de estados da ordenação obtida - Passo 4

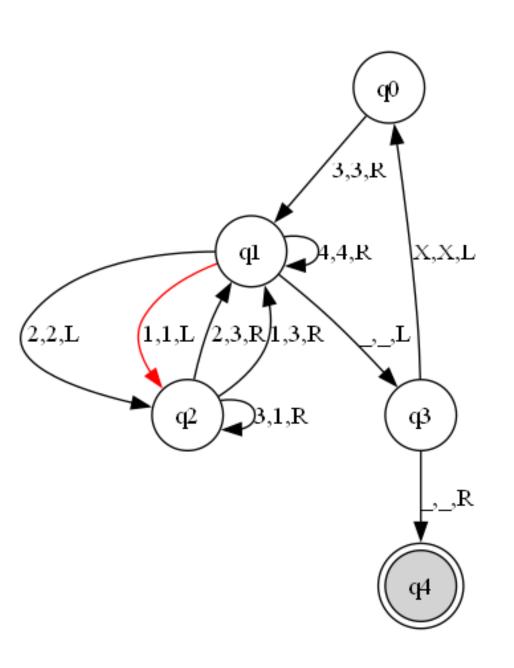


Figura 4.2 - Diagrama de estados da ordenação obtida - Passo 5

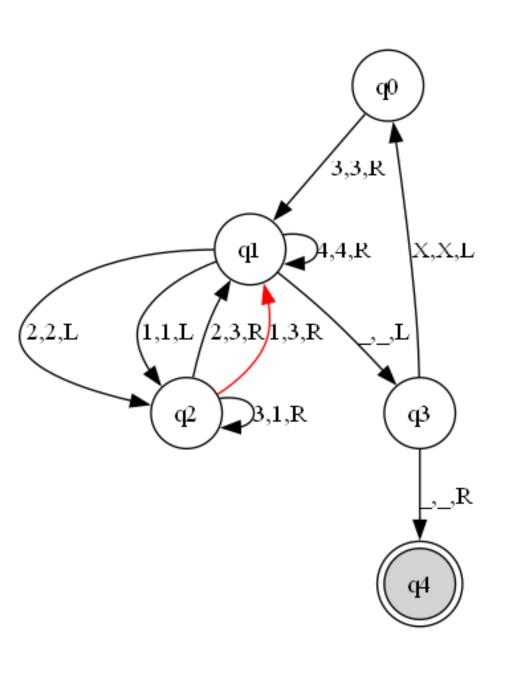


Figura 4.2 - Diagrama de estados da ordenação obtida - Passo 6

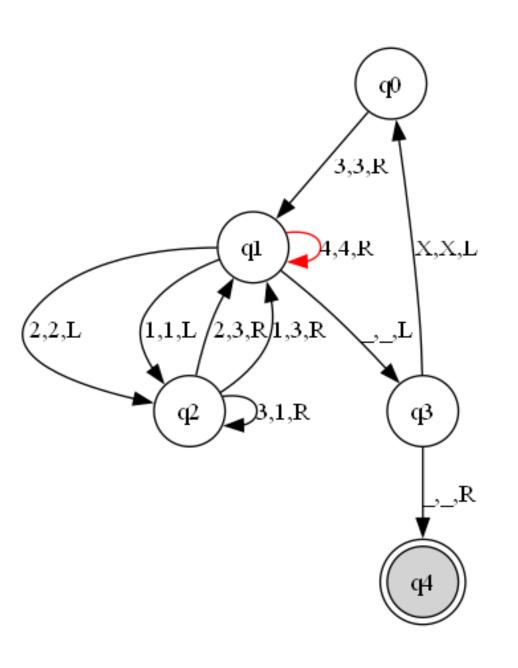


Figura 4.2 - Diagrama de estados da ordenação obtida - Passo 7

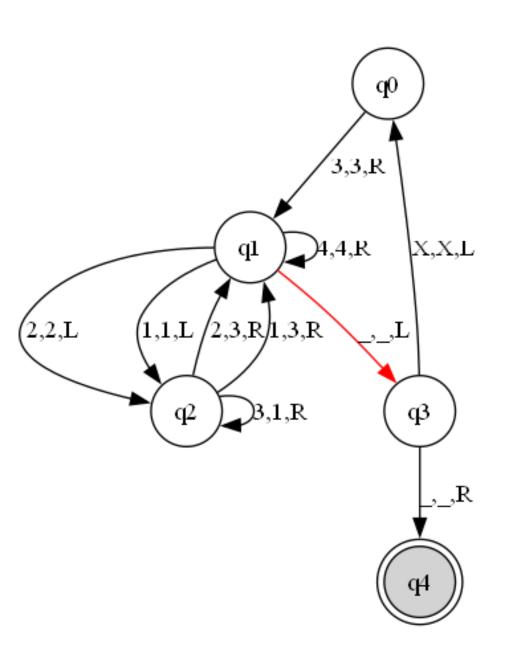


Figura 4.2 - Diagrama de estados da ordenação obtida - Passo 8

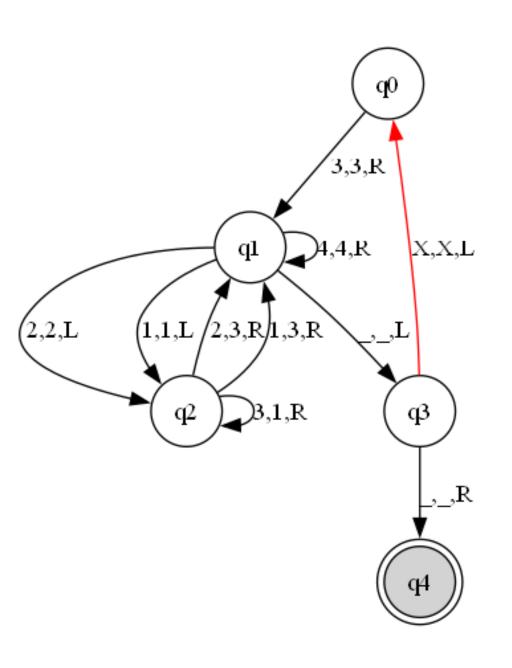


Figura 4.2 - Diagrama de estados da ordenação obtida - Passo 9

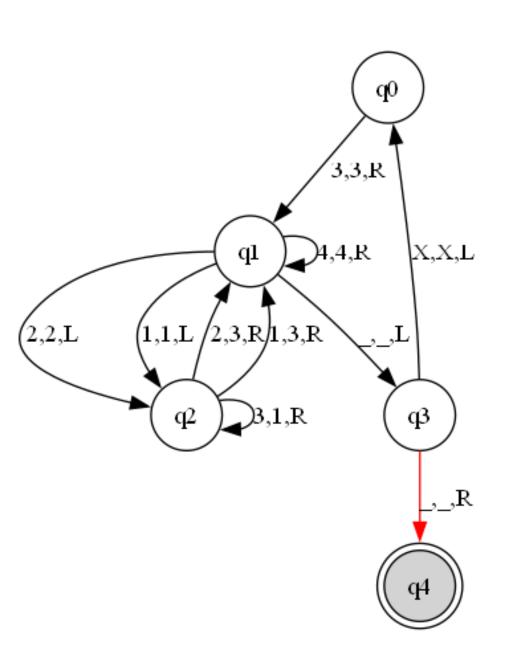


Figura 4.2 - Diagrama de estados da ordenação obtida - Passo 10

Estado Atual	Simbolo lido	Próximo estado	Simbolo escrito	Direção do movimento	Condição
q0	1	q1	1	Direita	Movimenta para o próximo
q1	4	q1	4	Direita	Compara com o próximo
q1	4	q1	4	Direita	Compara com o próximo
q1	3	q2	3	Esquerda	Troca para a esquerda
q2	3	q1	4	Direita	Troca para a direita
q1	2	q2	2	Esquerda	Troca
q2	4	q2	2	Direita	Troca para a esquerda
q2	2	q1	4	Direita	Troca para a direita
q1	_	q3	_	Esquerda	Fim da fita, checar flag de troca
q3	X	q0	Х	Esquerda	Reiniciar se houver alguma troca
q3	_	q4	_	-	Fim da ordenação

Tabela 4.3 - Tabela de transições para ordenação da lista (1,4,4,3,2)

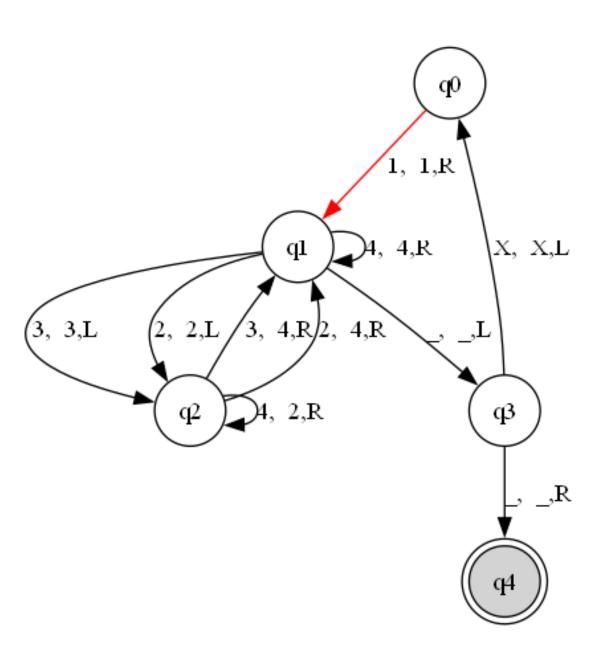


Figura 4.3 - Diagrama de estados da ordenação obtida -Passo 1

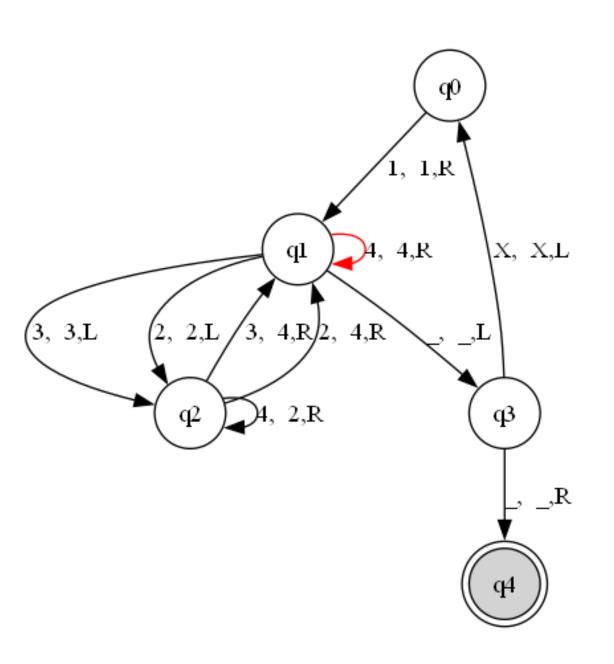


Figura 4.3 - Diagrama de estados da ordenação obtida -Passo 2

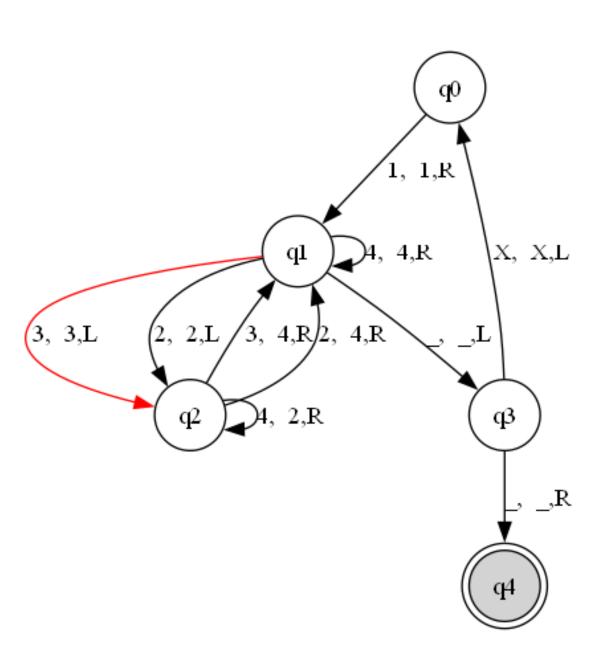


Figura 4.3 - Diagrama de estados da ordenação obtida -Passo 3

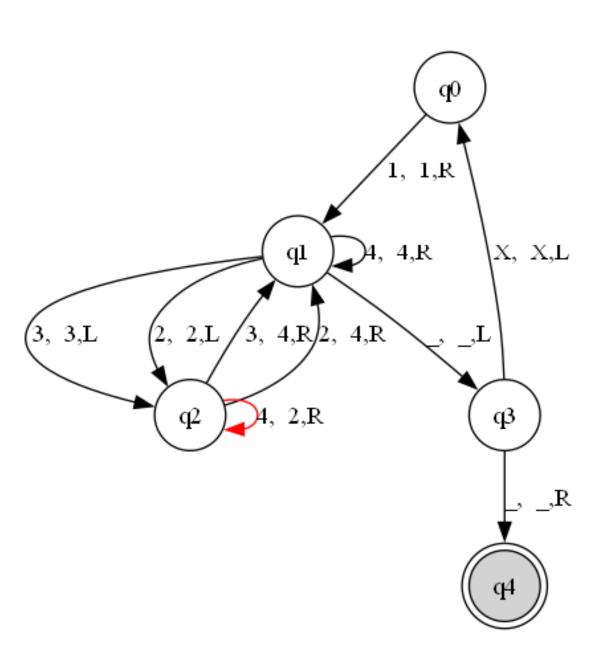


Figura 4.3 - Diagrama de estados da ordenação obtida -Passo 4

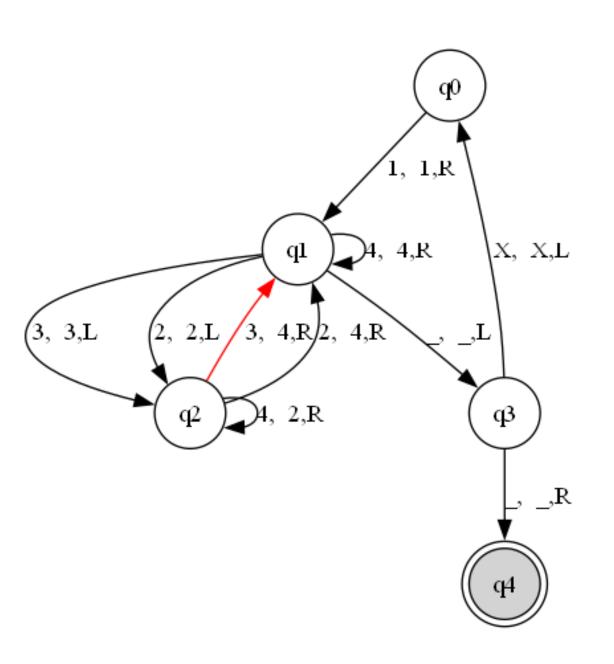


Figura 4.3 - Diagrama de estados da ordenação obtida -Passo 5

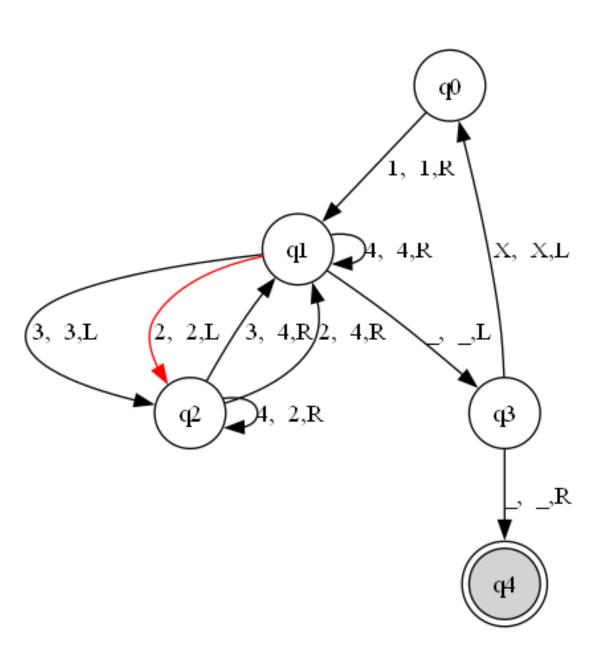


Figura 4.3 - Diagrama de estados da ordenação obtida -Passo 6

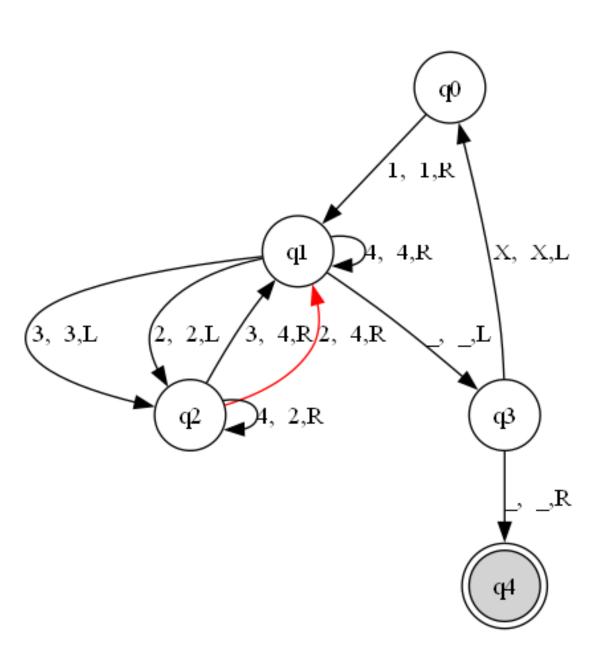


Figura 4.3 - Diagrama de estados da ordenação obtida -Passo 7

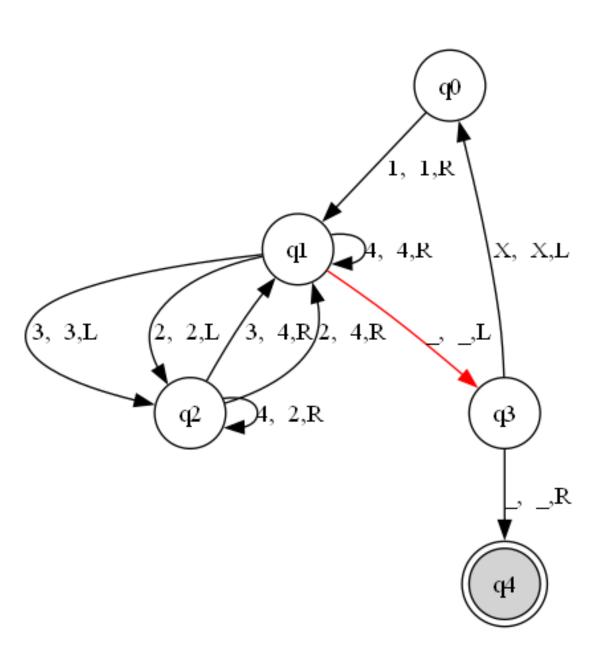


Figura 4.3 - Diagrama de estados da ordenação obtida -Passo 8

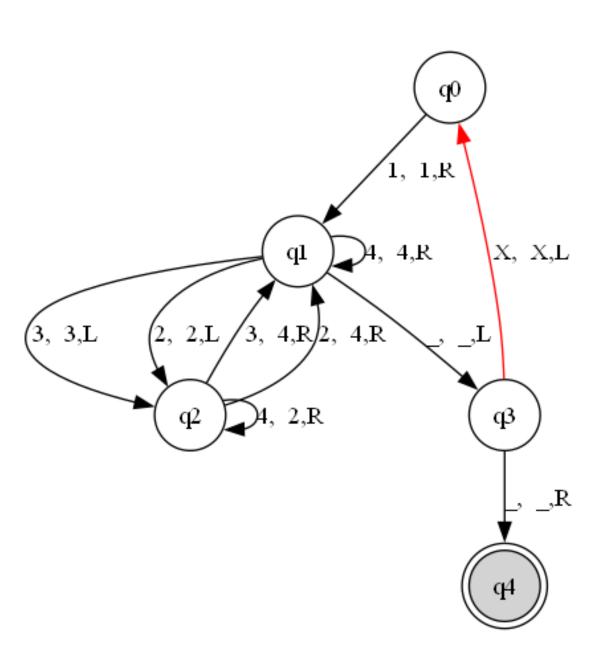


Figura 4.3 - Diagrama de estados da ordenação obtida -Passo 9

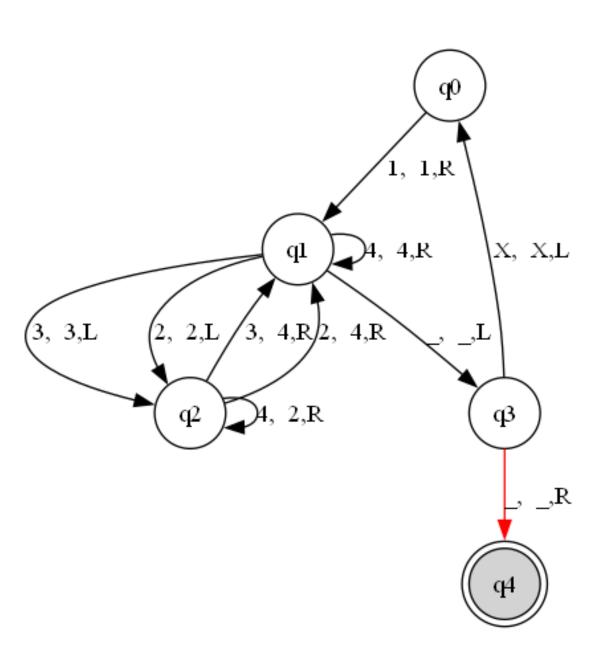


Figura 4.3 - Diagrama de estados da ordenação obtida -Passo 10

CONCLUSÃO

 Qualquer algoritmo, incluindo o Bubble Sort, pode ser executado por uma Máquina de Turing.

- Contribuição significativa
- Potencial para expansão

PROGRAMA EM FUNCIONAMENTO

REFERÊNCIAS

RAMOS, MARCUS V. M. LINGUAGENS FORMAIS: TEORIA, MODELAGEM E IMPLEMENTAÇÃO. 1ª ED. PORTO ALEGRE: BOOKMAN, 2009.

PAULO BLAUTH MENEZES. LINGUAGENS FORMAIS E AUTÔMATOS: VOLUME 3 DA SÉRIE LIVROS DIDÁTICOS INFORMÁTICA UFRGS. [S.L.] BOOKMAN EDITORA, 2009.

RMIN, WANG. ANALYSIS ON BUBBLE SORT ALGORITHM OPTIMIZATION. IN: 2010 INTERNATIONAL FORUM ON INFORMATION TECHNOLOGY AND APPLICATIONS. IEEE, 2010. P. 208-211.. ACESSO EM: 10 JAN. 2025.

REPRINTSEV, ALEX. TURING COMPLETENESS. IN: ORACLE SQL REVEALED: EXECUTING BUSINESS LOGIC IN THE DATABASE ENGINE. BERKELEY, CA: APRESS, 2018. P. 235-242.



IMPLEMENTAÇÃO DO ALGORITMO DE ORDENAÇÃO BUBBLE SORT UTILIZANDO MÁQUINA DE TURING

Orientador: Dr. Thales L. A. Valente

OBRIGADO

- FERNANDA SOUSA DE ASSUNÇÃO VALE
- JHONES DE SOUSA SOARES