Questão Desafio - MT 1

Dupla: Conrado Luiz e Luíza Guedes

Problema: Máquina te Turing que computa a soma de dois números binários, separados por o símbolo +.

1 - Descrição de alto nível

A nossa MT funciona em algumas etapas para conseguir computar a soma dos números binários. Tomando como exemplo a soma de 2 + 2, em binário 10 + 10, a máquina inicia com a seguinte fita:

$$q_0 10 + 10$$

1. Criar o espaço que será usado para computar o resultado. Nessa etapa, a MT soma a quantidade de algarismos dos dois números para garantirmos que não irá ocorrer um problema de overflow com a soma dos números. No final dessa etapa, a fita fica da seguinte forma:

$$B\ XY + Xq_4Y = EEEEB$$

2. A segunda etapa consiste em, ao mesmo tempo que transformamos os X em 1 e os Y em 0, colocamos o segundo número na parte da palavra que ficará o resultado (parte com os E). No final dessa etapa, a MT também converte os símbolos E restantes no final da fita para 0, para que possamos realizar as somas nele. Ao terminar essa etapa, a fita está conforme abaixo:

$$B\ XY + 10 = 001q_{12}0B$$

3. A terceira etapa é simples, precisamos voltar para o inicio da palavra para começar a computar a soma dos números. Nesse processo, também precisamos transformar de volta os X e Y em 1s e 0s do primeiro número, respectivamente. No final dessa etapa, a fita está dessa foram:

$$B\ q_{14} 10 + 10 = 0010B$$

4. A quarta etapa consiste do decremento do primeiro número. Para fazer essa computação, utiliza-se da propriedade do complemento de 2 do número binário. Primeiro o número binário é convertido para seu complemento de 2, depois a MT incrementa o complemento, após isso o complemento é retornado para o número, já com o decremento realizado. Após o processo dessa etapa, a fita se encontra da seguinte forma:

$$B\ 01 + q_{18} 10 = 0010B$$

5. quinta etapa, percorre a fita até a extremidade da direita para incrementar o resultado. Esse processo é feito trocando todos os 1s por 0s, da direita para esquerda, até encontrarmos um 0. Quando esse 0 é encontrado, trocamos ele por 1. Após terminar o processo de incremento, a MT retorna a cabeça da fita para a extremidade da esquerda, para repetir a etapa 4. Após o termino da etapa 5, a fica se encontra da seguinte forma:

$$B\ q_{14} 01 + 10 = 0011B$$

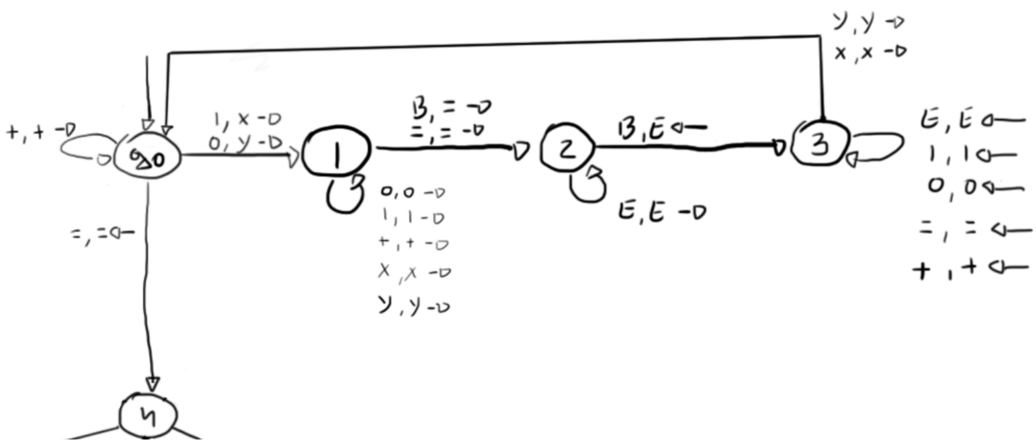
6. Após a MT repetir as etapas 4 e 5 até o complemento de 2 do primeiro número ser todo 1, ele entra na etapa 6. Nessa etapa, a MT percorre a fita da extremidade da esquerda até encontrar o símbolo =, após isso ela fazemos algumas verificações para saber onde está o início da resposta. Se a resposta for toda 0, a MT termina no estado q28. Se a resposta consistir de apenas um 1, a MT termina no estado q27. Para todos os outros casos, a MT termina no estado q24. Todos os términos deixam o ponteiro da fita no inicio da resposta, de forma que, se lermos a fita a partir de onde a MT deixou o ponteiro até a extremidade da direita, obteremos a resposta da soma.

$$B\,00 + 10 = 0q_{24}100B$$

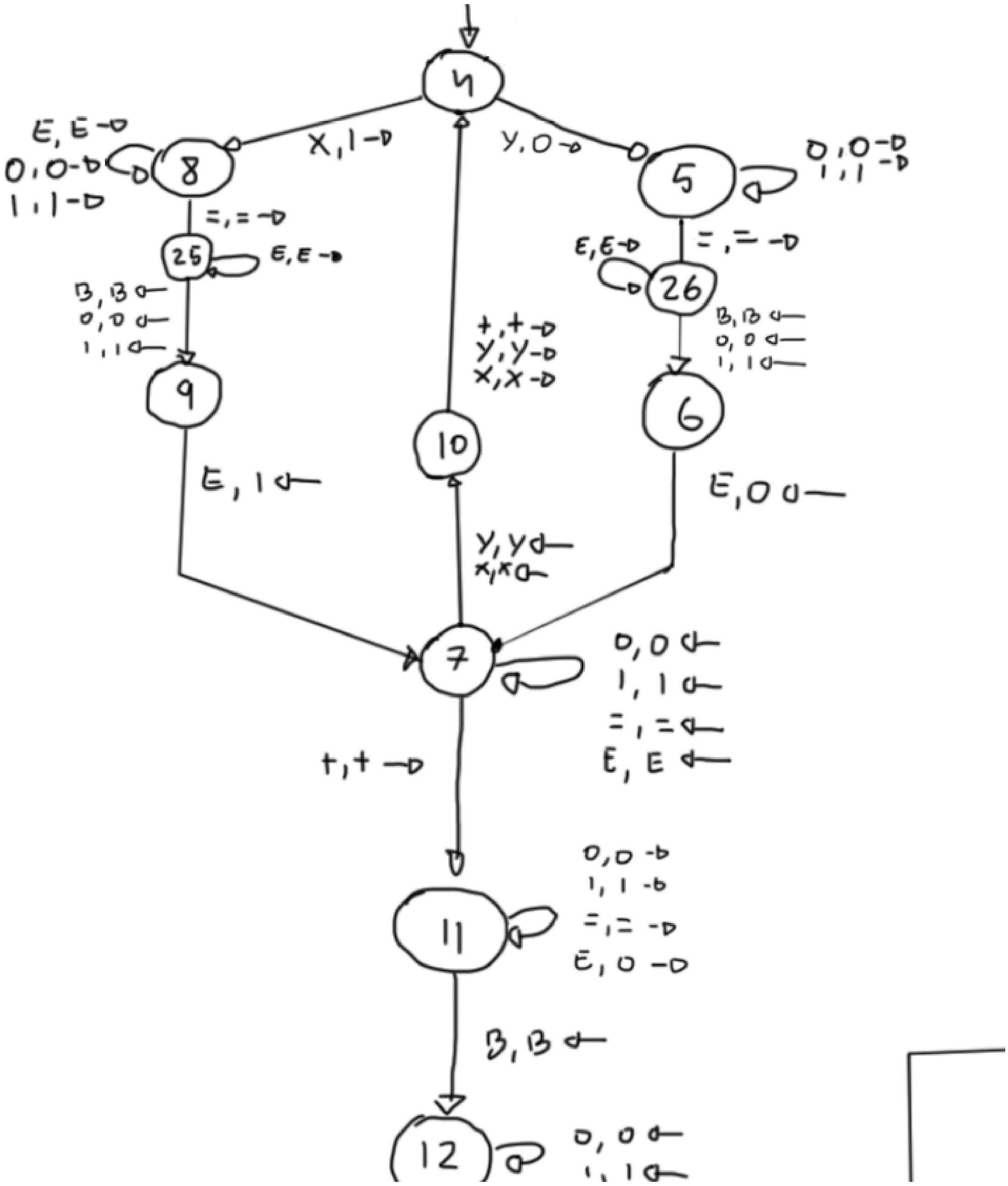
2 - Descrição formal

Nessa seção serão apresentadas as descrições formais de cada etapa, para melhor compreensão das etapas, e no final será apresentado a MT por completo.

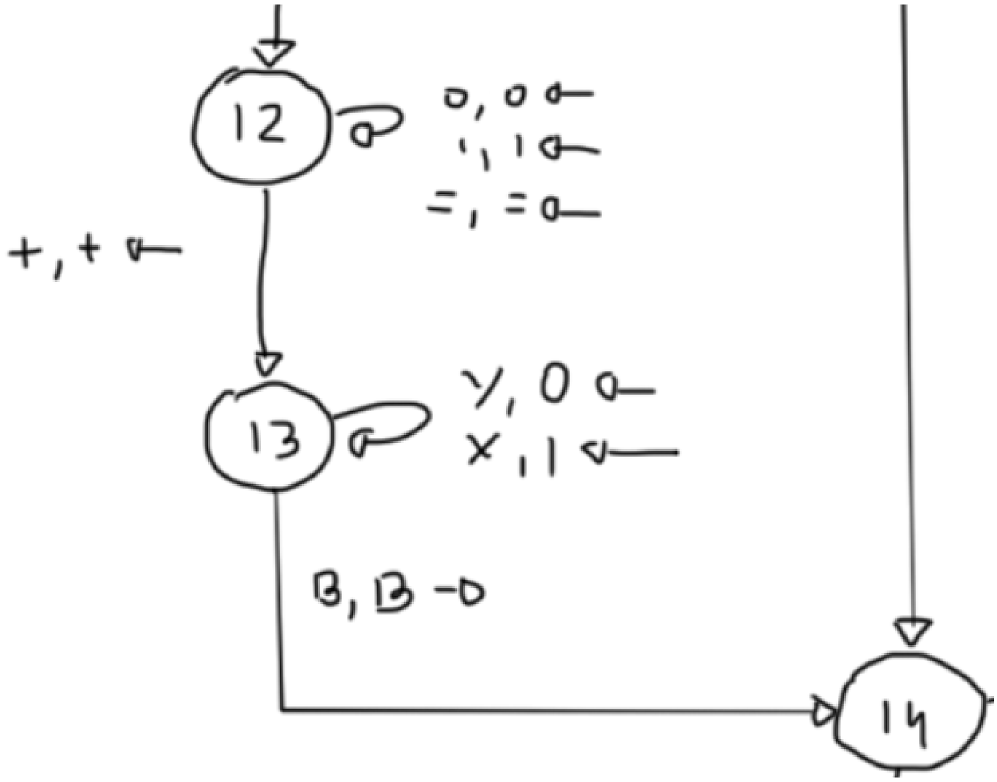
Etapa 1



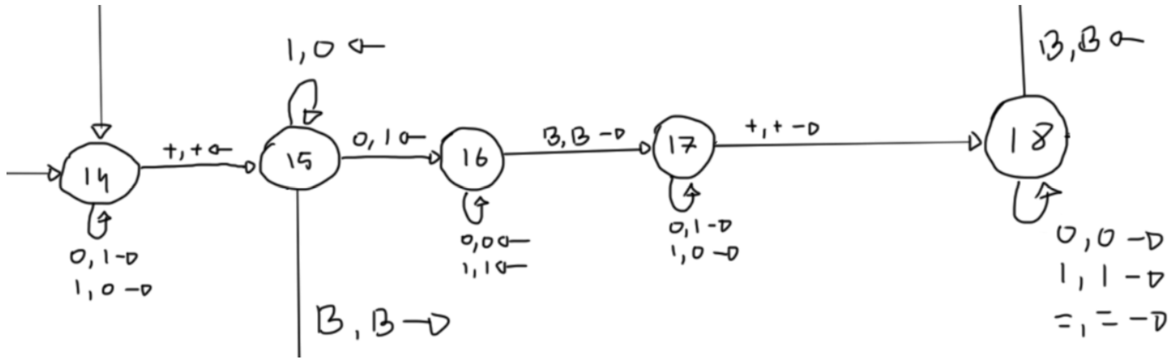
Etapa 2


4
X,1 →
Y,0 →
E,E →
0,0 →
1,1 →
8
=,= →
25
E,E →
B,B ←
0,0 ←
1,1 ←
9
E,1 ←
5
0,0 →
1,1 →
=,= →
E,E →
26
B,B ←
0,0 ←
1,1 ←
6
E,0 ←
+,+ →
Y,Y →
X,X →
10
Y,Y ←
X,X ←
7
0,0 ←
1,1 ←
=,= ←
E,E ←
+,+ →
11
0,0 →
1,1 →
=,= →
E,0 →
B,B ←
12
0,0 ←
1,1 ←

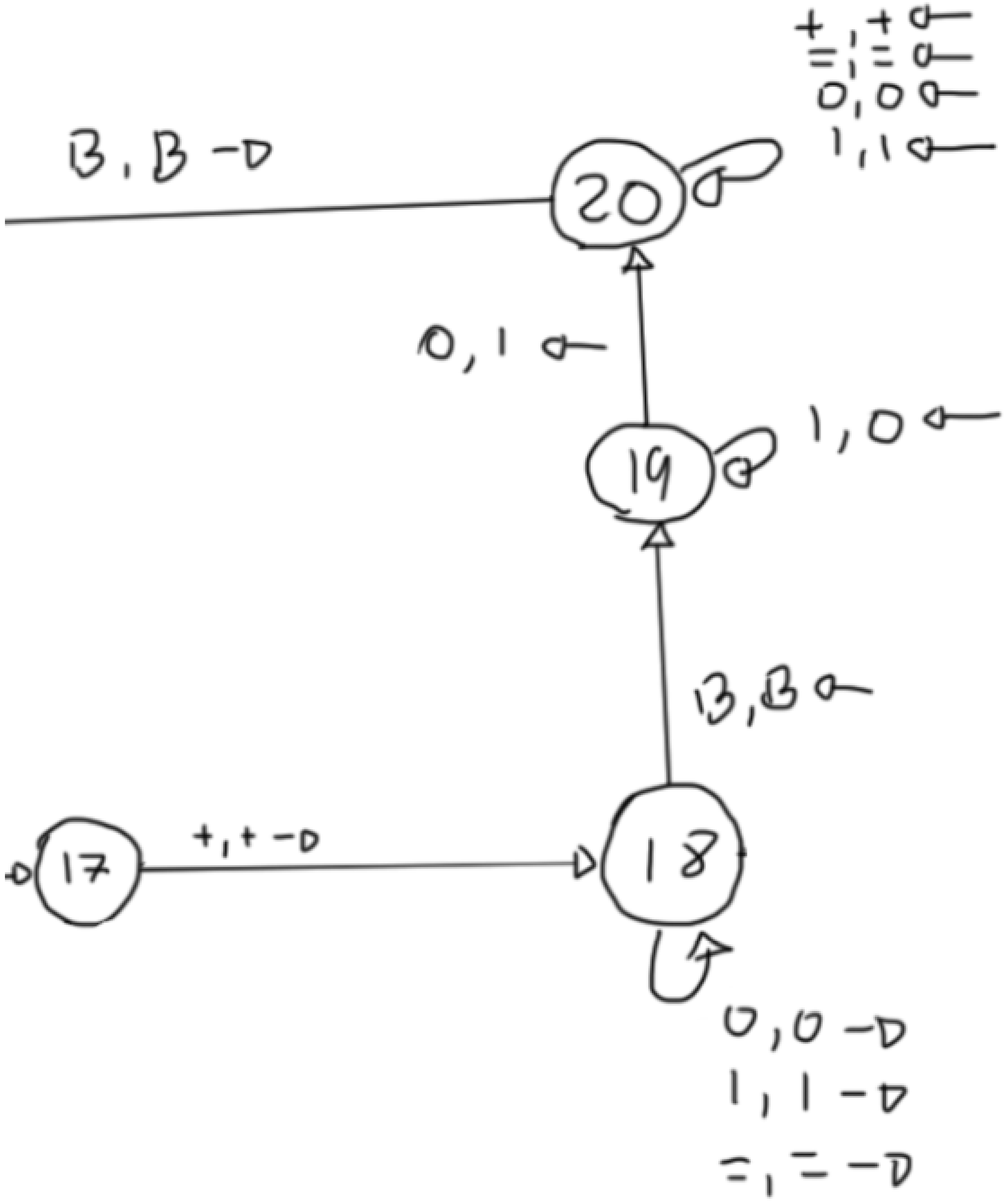
Etapa 3



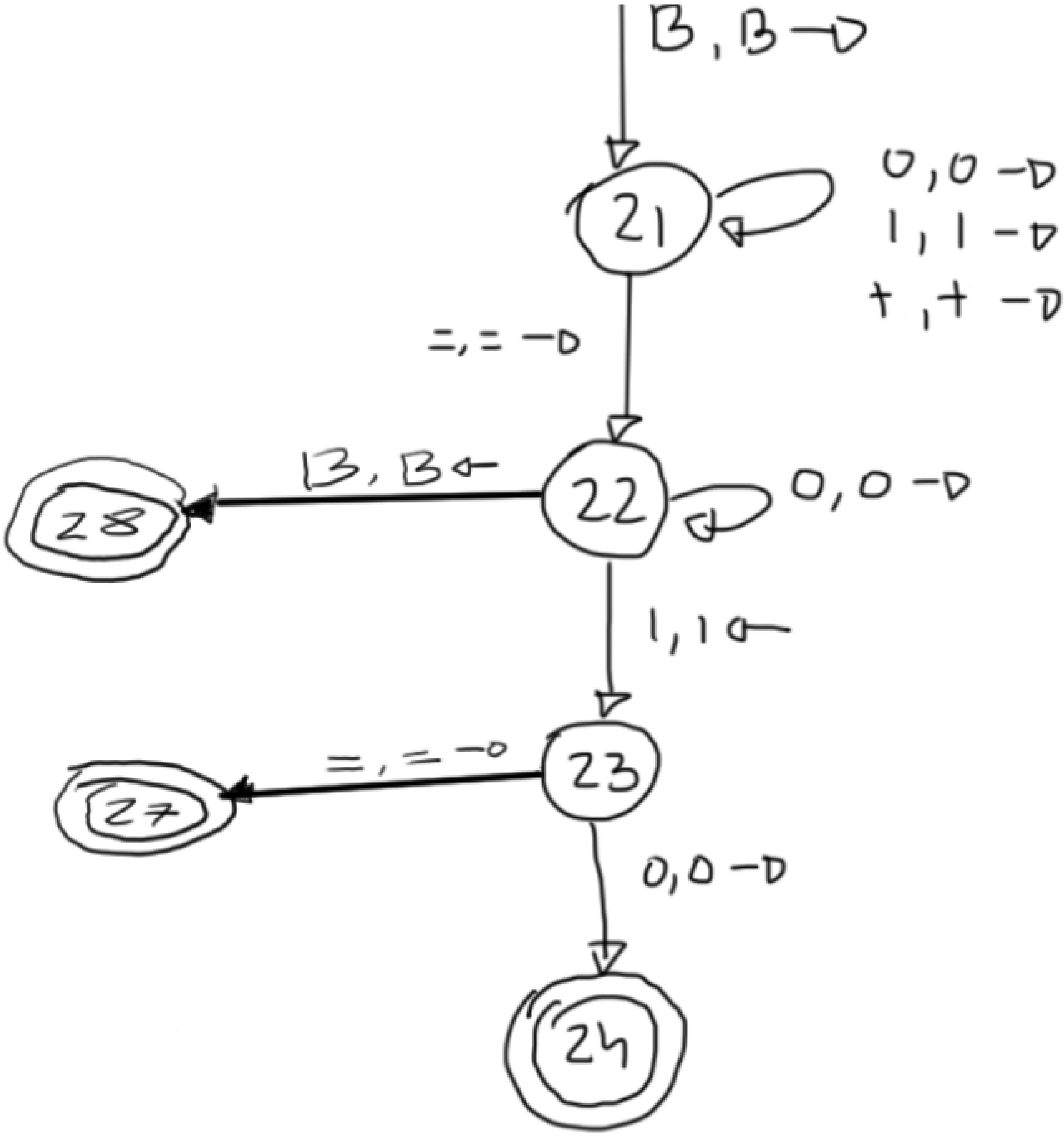
Etapa 4



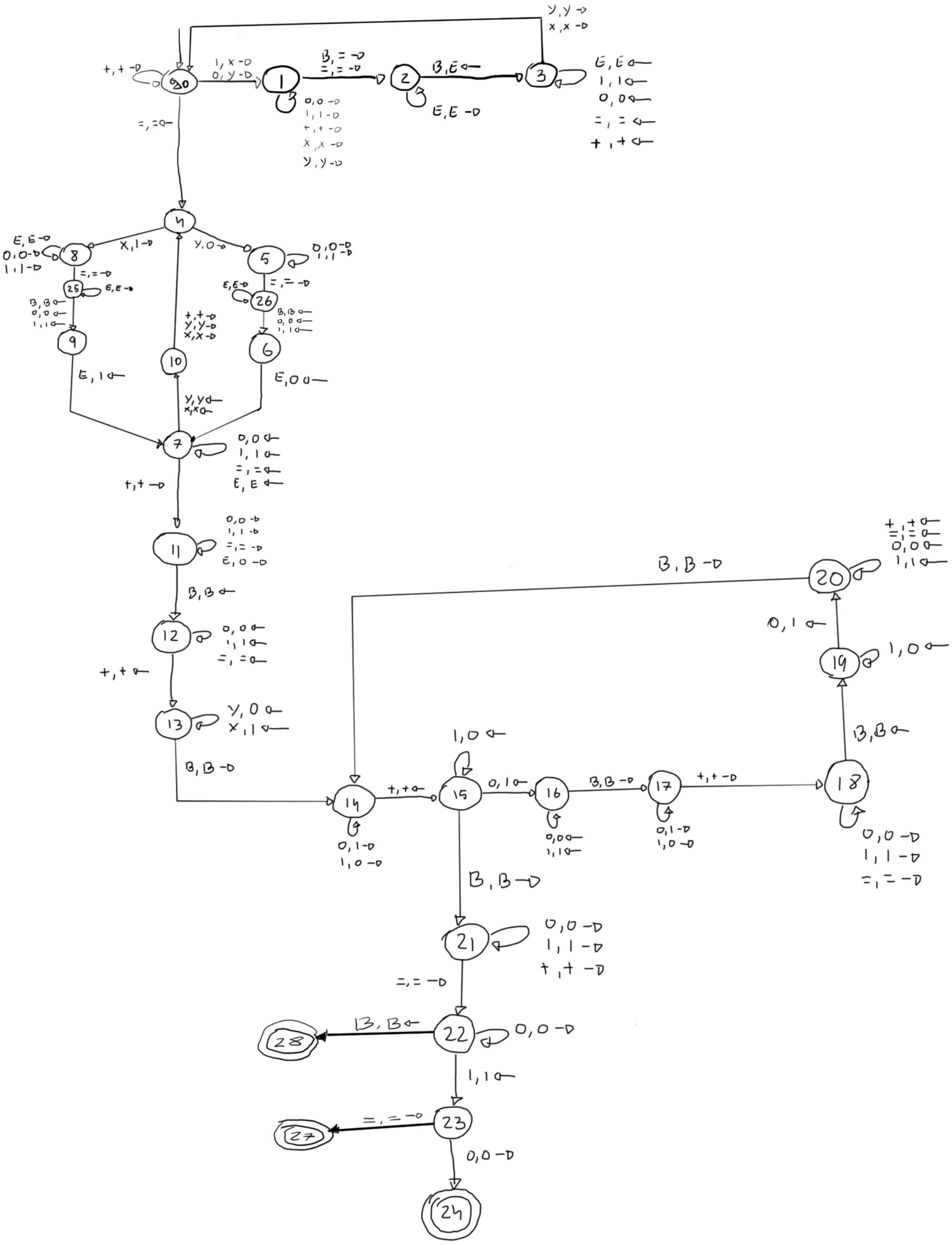
Etapa 5



Etapa 6



Maquina de Turing completa



Testes unitários

Como a natureza do problema é de fácil computação e teste, foi decidido fazer uma bateria de 10.000 testes automatizados, iterando sobre todas as combinações de números de 0 à 100. Para isso foi utilizado o código abaixo:

```
for i in range(100):
    for j in range(100):

        # Tranformando os indices para binário (tipo string)
        n1 = bin(i)
        n2 = bin(j)

        # Montando a fita inicial
        fita = [*n1[2:], '+', *n2[2:]]
```

```
        # Calculado o resultado esperado e transformando para binário
        result = bin(i + j)[2:]

        # Computando a resposta com a MT
        fita_resultante, ponteiro = movimento(M, fita)

        # Lendo a resposta da fita e juntando numa string para comparar
        result_mt = ''.join(read_strip_until_end(fita_resultante, ponteiro))

        # Criando a função de teste parametrizada
        test_name = f'test_{i}_{j}'
        exec(
            f"""def {test_name}(): assert {result} == {result_mt}"""
        )
```

Nesse código é feito um *nested loop* para iterar sobre todas as combinações de números entre [0,100). Os números são transformados em binário pela função *bin* do *python*, que retorna uma *string 0b10*, para um input do número 2 em base 10, por exemplo. Em seguida, a fita inicial é montada pegando só a parte do número retornado pela função *bin*, ou seja, excluindo o *0b* e pegando apenas *10*. Após feito isso, o resultado da soma dos dois números é computado e transformado em binário, da mesma forma que foi feita a construção da fita.

Com isso feito, foi utilizada a função de movimento da MT, que recebe a definição da MT e a fita inicial. A função de movimento retorna a fita resultante, com a resposta da soma computada pela MT e a posição na fita que o ponteiro foi deixado pela MT. Para obter a resposta limpa, sem os símbolos de fita vazia, etc, é utilizada a função *read_strip_until_end* que recebe como parâmetros a fita resultante da computação da MT e o índice que o ponteiro foi deixado. Essa função apenas lê a fita da posição que o ponteiro foi deixado pela MT até a extremidade da direita da fita, removendo os símbolos vazios. O retorno dessa função é um *array* com cada posição contendo um símbolo da resposta. Para transformar o *array* numa *string*, para ser comparada a resposta esperada, os símbolos do *array* são juntados em uma *string*. Com a resposta esperada, e a resposta da MT, uma função parametrizada é criada para ser passiva de teste, pelo módulo *pytest*.

O código acima foi rodado conforme explicitado abaixo, com o pipe para um arquivo *tests.txt* para salvar o output do *pytest*:

```
py -m pytest .\unit.py > tests.txt
```

O código acima produz o arquivo *tests.txt* abaixo, com todos os testes passando:

```
============================= test session starts =============================
platform win32 -- Python 3.7.9, pytest-6.2.3, py-1.10.0, pluggy-0.13.1
rootdir: C:\Users\Conrado\Documents\Dev\Teoria-da-Computacao\extra
collected 10000 items

unit.py ................................................................ [  0%]
........................................................................ [  1%]
........................................................................ [  2%]
........................................................................ [  2%]
........................................................................ [  3%]
........................................................................ [  4%]
........................................................................ [  4%]
........................................................................ [  5%]
```

```
............................................................................... [  6%]
............................................................................... [  7%]
............................................................................... [  7%]
............................................................................... [  8%]
............................................................................... [  9%]
............................................................................... [ 10%]
............................................................................... [ 10%]
............................................................................... [ 11%]
............................................................................... [ 12%]
............................................................................... [ 12%]
............................................................................... [ 13%]
............................................................................... [ 14%]
............................................................................... [ 15%]
............................................................................... [ 15%]
............................................................................... [ 16%]
............................................................................... [ 17%]
............................................................................... [ 17%]
............................................................................... [ 18%]
............................................................................... [ 19%]
............................................................................... [ 20%]
............................................................................... [ 20%]
............................................................................... [ 21%]
............................................................................... [ 22%]
............................................................................... [ 22%]
............................................................................... [ 23%]
............................................................................... [ 24%]
............................................................................... [ 25%]
............................................................................... [ 25%]
............................................................................... [ 26%]
............................................................................... [ 27%]
............................................................................... [ 28%]
............................................................................... [ 28%]
............................................................................... [ 29%]
............................................................................... [ 30%]
............................................................................... [ 30%]
............................................................................... [ 31%]
............................................................................... [ 32%]
............................................................................... [ 33%]
............................................................................... [ 33%]
............................................................................... [ 34%]
............................................................................... [ 35%]
............................................................................... [ 35%]
............................................................................... [ 36%]
............................................................................... [ 37%]
............................................................................... [ 38%]
............................................................................... [ 38%]
............................................................................... [ 39%]
............................................................................... [ 40%]
............................................................................... [ 40%]
............................................................................... [ 41%]
............................................................................... [ 42%]
............................................................................... [ 43%]
............................................................................... [ 43%]
............................................................................... [ 44%]
............................................................................... [ 45%]
............................................................................... [ 46%]
............................................................................... [ 46%]
............................................................................... [ 47%]
```

```
........................................................................ [ 48%]
........................................................................ [ 48%]
........................................................................ [ 49%]
........................................................................ [ 50%]
........................................................................ [ 51%]
........................................................................ [ 51%]
........................................................................ [ 52%]
........................................................................ [ 53%]
........................................................................ [ 53%]
........................................................................ [ 54%]
........................................................................ [ 55%]
........................................................................ [ 56%]
........................................................................ [ 56%]
........................................................................ [ 57%]
........................................................................ [ 58%]
........................................................................ [ 58%]
........................................................................ [ 59%]
........................................................................ [ 60%]
........................................................................ [ 61%]
........................................................................ [ 61%]
........................................................................ [ 62%]
........................................................................ [ 63%]
........................................................................ [ 64%]
........................................................................ [ 64%]
........................................................................ [ 65%]
........................................................................ [ 66%]
........................................................................ [ 66%]
........................................................................ [ 67%]
........................................................................ [ 68%]
........................................................................ [ 69%]
........................................................................ [ 69%]
........................................................................ [ 70%]
........................................................................ [ 71%]
........................................................................ [ 71%]
........................................................................ [ 72%]
........................................................................ [ 73%]
........................................................................ [ 74%]
........................................................................ [ 74%]
........................................................................ [ 75%]
........................................................................ [ 76%]
........................................................................ [ 76%]
........................................................................ [ 77%]
........................................................................ [ 78%]
........................................................................ [ 79%]
........................................................................ [ 79%]
........................................................................ [ 80%]
........................................................................ [ 81%]
........................................................................ [ 82%]
........................................................................ [ 82%]
........................................................................ [ 83%]
........................................................................ [ 84%]
........................................................................ [ 84%]
........................................................................ [ 85%]
........................................................................ [ 86%]
........................................................................ [ 87%]
........................................................................ [ 87%]
........................................................................ [ 88%]
........................................................................ [ 89%]
```

```
........................................................................ [ 89%]
........................................................................ [ 90%]
........................................................................ [ 91%]
........................................................................ [ 92%]
........................................................................ [ 92%]
........................................................................ [ 93%]
........................................................................ [ 94%]
........................................................................ [ 94%]
........................................................................ [ 95%]
........................................................................ [ 96%]
........................................................................ [ 97%]
........................................................................ [ 97%]
........................................................................ [ 98%]
........................................................................ [ 99%]
........................................................................ [100%]

====================== 10000 passed in 67.32s (0:01:07) =======================
```

Métricas e Complexidade temporal

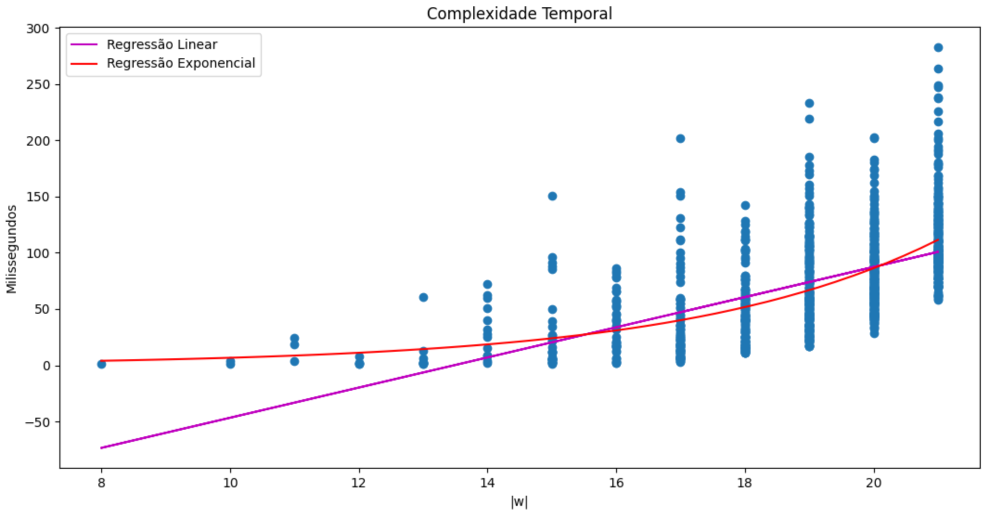
Para verificar a complexidade temporal da MT, foi feita uma amostra aleatória de tamanho 1000, pegando números de 0 à 1000, transformando em binário e computando a soma pela MT. O código abaixo foi utilizado:

```
np.random.seed(10)

    n_sucesso = 0
    len_palavras = []
    timings = []

    for _ in range(1000):

        i = np.random.randint(0, 1000)
        j = np.random.randint(0, 1000)

        n1 = bin(i)
        n2 = bin(j)
        fita = [*n1[2:], '+', *n2[2:]]
        len_palavras.append(len(fita))
        result = bin(i + j)[2:]
        print(i, j)
        ts = time()
        fita_resultante, ponteiro = movimento(M, fita)
        te = time()

        timings.append((te-ts) * 1000)

        result_mt = ''.join(read_strip_until_end(fita_resultante, ponteiro))

    X = sm.add_constant(len_palavras)
    model = sm.OLS(timings, X)
    result = model.fit()

    regression_line = [result.params[0] + result.params[1]*len_palavra for
len_palavra in len_palavras]
```

```
exp, _ = curve_fit(lambda t, a, b: a*np.exp(b*t), len_palavras, timings)

exp_x = np.linspace(min(len_palavras), max(len_palavras))
exponential_line = [exp[0]*np.exp(exp[1]*x) for x in exp_x]

fig, ax = plt.subplots()

ax.plot(len_palavras, regression_line, c='m', label='Regressão Linear')
ax.plot(exp_x, exponential_line, c='r', label='Regressão Exponencial')

ax.scatter(len_palavras, timings)
ax.legend()
ax.set_title('Complexidade Temporal')
ax.set_xlabel('|w|')
ax.set_ylabel('Milissegundos')
plt.show()
```

Para construção do gráfico, a biblioteca *matplotlib* foi utilizada. As bibliotecas *statsmodels* e *scipy* foram utilizadas para fazer as linhas de tendência, regressão linear e regressão exponencial. Foi utilizado um seed de 10 pelo *numpy* para que os resultados sejam reproduzíveis.

O gráfico de complexidade da MT foi gerado:



É possível observar que a regressão exponencial captura melhor o comportamento da MT. A medida que o tamanho da palavra cresce, o tempo de computação da MT aumenta de forma exponencial.