

Universidad de Buenos Aires (FIUBA)
Maestría en Inteligencia Artificial

Trabajo Práctico 1

Máquina de Turing: suma de dos números en binario

Materia: Computación, Algoritmos y Estructuras de Datos
Docente: Dr. Lic. Camilo Argoty
Alumna: Esp. Lic. Noelia Qualindi
SIU: a1411

Repositorio del trabajo:
https://github.com/noequalindi/computing_algorithms/tree/main/tp1

1. Enunciado elegido

Según el último dígito del SIU (a1411), corresponde el punto 1 del TP1: *suma de dos números en binario*.

2. Formato y supuestos de entrada

Se implementa una Máquina de Turing (MT) de 2 cintas (multi-tape) definida en un archivo YAML:

- Cinta 1: número binario A
- Cinta 2: número binario B (también se usa para escribir el resultado)

La MT opera en *little-endian* (bit menos significativo primero). Por eso, el runner invierte las cadenas ingresadas en binario normal (big-endian) antes de cargar la cinta, y vuelve a invertir el resultado al finalizar.

3. Idea de la máquina

La suma binaria se modela con dos “modos” (estados):

- **no carry**: no hay acarreo.
- **carry**: hay acarreo.

La máquina recorre los bits desde el menos significativo hacia la izquierda, escribiendo el bit suma en la cinta 2 y actualizando el estado según el acarreo.

4. Archivos entregados

- `tp1/binary_addition.yaml`: definición de la MT.
- `tp1/tp1_run_tm.py`: runner para ejecutar y producir evidencia.

5. Cómo correr (evidencia)

```
cd tp1
python tp1_run_tm.py --a 1011 --b 111
```

La salida incluye:

- resultado en binario (big-endian),
- tapes finales (útil para captura),
- cantidad de pasos.

6. Evidencia de ejecución (local)

Se ejecutó el script `tp1_run_tm.py` con tres casos de prueba.

6.1. Caso 1: $1 + 1$

```
python tp1_run_tm.py --a 1 --b 1
```

```
Entrada A (bin): 1
Entrada B (bin): 1
Resultado (bin): 10
Tape1 final (trim): 1
Tape2 final (trim): 10
```

6.2. Caso 2: $1111 + 1$

```
python tp1_run_tm.py --a 1111 --b 1
```

```
Entrada A (bin): 1111
Entrada B (bin): 1
Resultado (bin): 10000
Tape1 final (trim): 1111
Tape2 final (trim): 10000
```

6.3. Caso 3: $0 + 0$

```
python tp1_run_tm.py --a 0 --b 0
```

```
Entrada A (bin): 0
Entrada B (bin): 0
Resultado (bin): 0
Tape1 final (trim): 0
Tape2 final (trim): 0
```

6.4. Caso 4: $1101 + 101$

En el simulador en línea este caso se ingresa como 1101#101. En la ejecución local se pasan los operandos por parámetros separados.

```
python tp1_run_tm.py --a 1101 --b 101
```

```
=====
Entrada A (bin): 1101
Entrada B (bin): 101
Resultado (bin): 10010
Pasos: 10
Tape1 final (trim): 1101
Tape2 final (trim): 10010
=====
```

7. Comprobación en simulador en línea

La máquina de Turing fue validada en un simulador web. Se adjunta evidencia visual de la ejecución y un enlace compartible que permite reproducir la comprobación.

7.1. Simulador y enlace

- Simulador: <https://turingmachinesimulator.com/>

- Enlace (Share Link) de la máquina validada: <http://turingmachinesimulator.com/shared/yoxceanwzp>

Caso 1101#101 : el siguiente caso se utilizó tanto para la validación en el simulador en línea como para la ejecución local.

- Entrada en simulador online: 1101#101
- Entrada en ejecución local: -a 1101 -b 101
- Salida obtenida: 10010

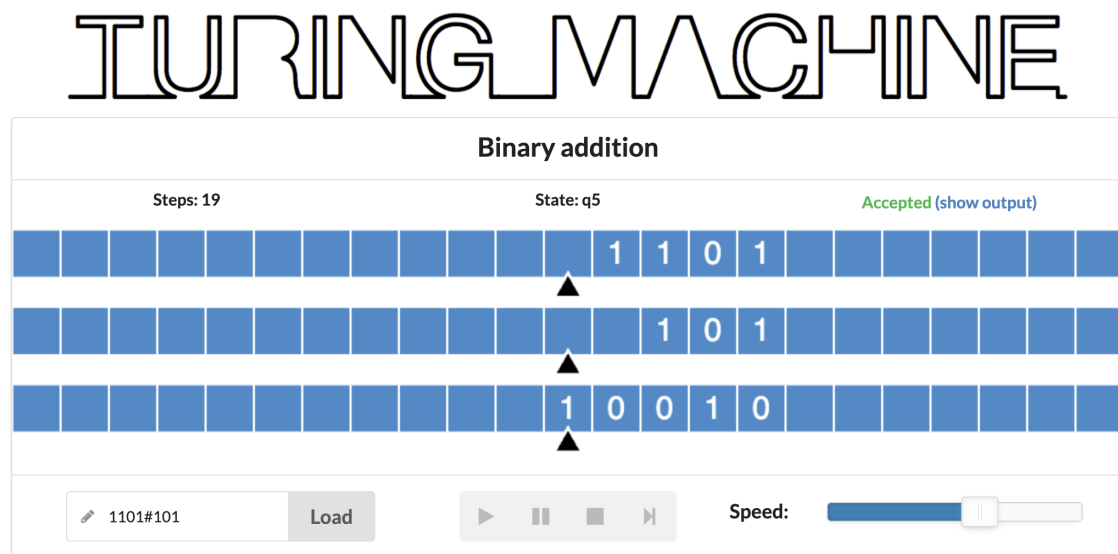


Figura 1: Comprobación en simulador en línea de la suma binaria con entrada 1101#101.

En la Figura 1 se observa el estado final de la ejecución, donde la cinta de salida contiene el resultado correcto de la suma.

8. Repositorio

El código fuente y evidencia se entregan vía repositorio:

https://github.com/noeliaqualindi/computing_algorithms

9. Bibliografía

Referencias

- [1] Michael Sipser. *Introduction to the Theory of Computation*. Cengage Learning, 3rd edition, 2012.
- [2] John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Pearson, 3rd edition, 2006.