Universidad del Valle de Guatemala

Facultad de Ingeniería

Análisis y Diseño de Algoritmos

PROYECTO 1

Gabriel Paz 221087

17 de febrero del 2025, Guatemala de la Asunción

Link de Repositorio en GitHub:
https://github.com/gabrielpaz2003/TuringMachineFibonacciP1.git

Link de Video Explicativo:

https://youtu.be/W1qG1bdNYDM

1. Introducción

Este informe documenta el desarrollo y análisis de una Máquina de Turing diseñada para calcular la sucesión de Fibonacci. Se investigó la notación asintótica O, se analizó el tiempo de ejecución de la máquina y se realizó una simulación para evaluar su comportamiento y eficiencia.
El proyecto incluye la implementación en Python de una Máquina de Turing determinista de una cinta, donde se define una convención para representar y manipular números naturales en la cinta.

2. Convenciones Elegidas

Para la implementación de la Máquina de Turing, se definieron las siguientes convenciones:

- **Representación de los números en la cinta:**
  Los números naturales se representan mediante una secuencia de **1s** en la cinta.
  Por ejemplo:
  - 111 se representa como 1
  - 222 se representa como 11
  - 333 se representa como 111, y así sucesivamente.
- **Blanco de la cinta:**
  Se usa el símbolo B para indicar celdas vacías de la cinta.
- **Estados de la máquina:**
  Se definieron diferentes estados (q0, q101, q102, etc.) que controlan la ejecución del algoritmo en la máquina.
- **Movimiento de la cabeza lectora:**
  La cabeza de lectura/escritura puede moverse a la **derecha (R)**, **izquierda (L)** o mantenerse en la misma celda (**N**).
- **Símbolos de operación:**
  La máquina usa los siguientes símbolos para marcar posiciones clave en la cinta:
  - X: Para marcar la posición inicial del cálculo.

- *: Marcador temporal en la secuencia de Fibonacci.
- B: Indica una celda vacía.

3. Diagrama de la Máquina de Turing

A continuación, se muestra el diagrama de estados de la Máquina de Turing implementada en este proyecto. Este diagrama visualiza los estados y transiciones que permiten calcular la sucesión de Fibonacci.

**Imagen con mejor resolución dentro del repositorio*

4. Implementación del Programa en Python

El código desarrollado consta de dos archivos principales:

- **Turing_Machine.py**: Contiene la implementación de la Máquina de Turing, la representación de la cinta y la lógica de simulación.
- **simulator.py**: Define la interfaz de usuario, permite ingresar un número para calcular su valor en la sucesión de Fibonacci y genera el análisis empírico del rendimiento.

Características del programa:

Se utilizan 2 archivos con formato .txt para la configuración de la Maquina de Turing:

turing_rules.txt y **turing_setup.txt**

5. Análisis Empírico

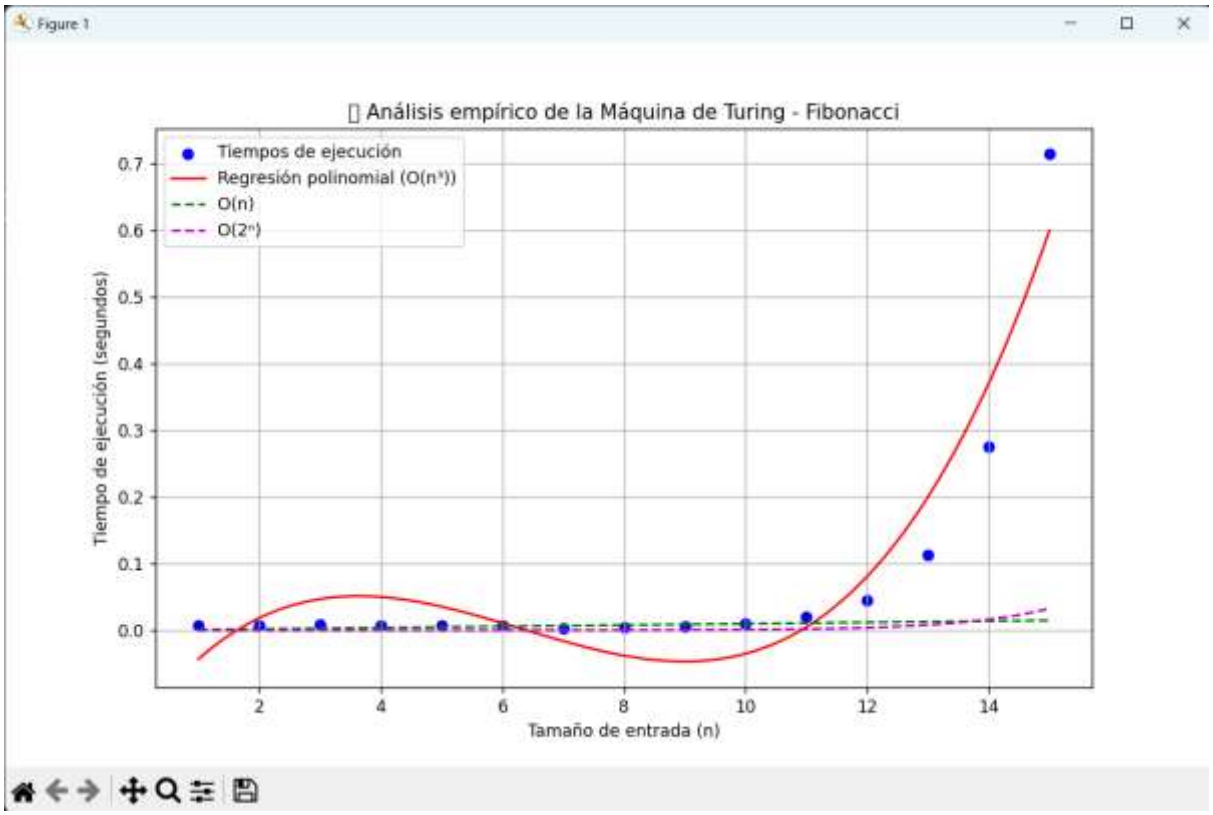
Para evaluar el rendimiento de la Máquina de Turing, se realizaron mediciones del tiempo de ejecución para diferentes valores de entrada n. A continuación, se presentan los resultados.

5.1 Listado de tiempos de ejecución

```
📊 Iniciando análisis empírico de la ejecución...

🔢 n= 1 | Fibonacci( 1)=   1 | ⏳ Tiempo=0.008491 s
🔢 n= 2 | Fibonacci( 2)=   1 | ⏳ Tiempo=0.010022 s
🔢 n= 3 | Fibonacci( 3)=   2 | ⏳ Tiempo=0.012538 s
🔢 n= 4 | Fibonacci( 4)=   3 | ⏳ Tiempo=0.008664 s
🔢 n= 5 | Fibonacci( 5)=   5 | ⏳ Tiempo=0.008505 s
🔢 n= 6 | Fibonacci( 6)=   8 | ⏳ Tiempo=0.008681 s
🔢 n= 7 | Fibonacci( 7)=  13 | ⏳ Tiempo=0.005116 s
🔢 n= 8 | Fibonacci( 8)=  21 | ⏳ Tiempo=0.005906 s
🔢 n= 9 | Fibonacci( 9)=  34 | ⏳ Tiempo=0.006644 s
🔢 n=10 | Fibonacci(10)=  55 | ⏳ Tiempo=0.012531 s
🔢 n=11 | Fibonacci(11)=  89 | ⏳ Tiempo=0.026790 s
🔢 n=12 | Fibonacci(12)= 144 | ⏳ Tiempo=0.059596 s
🔢 n=13 | Fibonacci(13)= 233 | ⏳ Tiempo=0.148924 s
🔢 n=14 | Fibonacci(14)= 377 | ⏳ Tiempo=0.322092 s
🔢 n=15 | Fibonacci(15)= 610 | ⏳ Tiempo=0.889842 s
```

5.2 Gráfico de dispersión de tiempos de ejecución



5.3 Regresión polinomial

Se ajustó una regresión polinomial de **O(n^3)** para modelar la tendencia del tiempo de ejecución. Se compararon diferentes órdenes de complejidad:

- O **Regresión polinomial O(n^3)** (Curva roja en la gráfica).

- O **O(n)** (Curva verde discontinua).
- O **O(2^n)** (Curva morada discontinua).

El ajuste sugiere que el tiempo de ejecución crece a un ritmo mayor que $O(n^2)$, acercándose a un crecimiento exponencial en valores más grandes de n.

6. Conclusiones

1. Se logró implementar una Máquina de Turing funcional capaz de calcular la sucesión de Fibonacci utilizando una representación en cinta basada en 1s.
2. El análisis empírico mostró que el tiempo de ejecución crece significativamente con n, evidenciando un posible comportamiento de tipo exponencial.
3. El análisis empírico mostró que el tiempo de ejecución crece considerablemente con n, indicando una posible complejidad mayor a O(n^2), con un comportamiento más cercano a O(2^n).
4. Se implementó una interfaz en consola optimizada y se agregaron detalles visuales para mejorar la experiencia del usuario.

8. Referencias

Wikipedia: Sucesión de Fibonacci