Maquina de Turing y su Relación con el Problema del Milenio “P=NP”

Merino Vidal Mateo Alejandro

Departamento de Informática,
Universidad Mayor de San Simón,
Cochabamba, Bolivia

202301308@est.umss.edu

***Abstract*—El presente articulo demuestra los conceptos fundamentales y desarrollos realizados con respecto a la Maquina de Turing, centrándose en su modelo determinista(DTM) y no determinista(NTM). Se detalla la historia de como surgieron estas maquinas como también su comportamiento y las descripciones de cada variante. Además, se demuestra como estos modelos llegan a relacionarse con el problema del milenio sobre como los problemas P son iguales a los problemas NP, que implica esto y si es una teoría valida o no en base a diversos fundamentos.**

I. Introducción

La computación es una disciplina que se encarga del estudio de la resolución de problemas a través de una herramienta denominada computadora.

Sin embargo, esta área estuvo en constante evolución a lo largo de la historia, desarrollándose en base a diversas aportaciones, elaboradas por notables personalidades.
Uno de los mas importantes conceptos fue la Maquina de Turing, siendo un modelo teórico diseñado en 1936 por Alan Turing, que ayudo a desarrollar una mayor comprensión sobre el concepto de computación y algoritmo.

En esencia era una maquina programable capaz de resolver cualquier problema que pudiera ser descrito en base a un conjunto de pasos o instrucciones denominado algoritmo.
Se considera a esta misma como el precursor histórico de los modernos computadores, destacando mucho por su sencillez y sus grandiosas posibilidades en cuanto a sus configuraciones.

Años después, con la finalidad de comprender aun mas a profundidad los límites en torno a que pueden calcular las maquinas, Alan Turing desarrollo un nuevo modelo de la Maquina de Turing denominado Maquina de Turing No Determinista(NTM), nombrando al modelo original como Maquina de Turing Determinista(DTM).

Ambos modelos permitieron desarrollar la teoría de la complejidad computacional, teniendo una relación estrecha con el problema del milenio “P=NP”, el cual menciona que si todo problema para el cual puede verificarse una solución rápidamente, también puede ser resuelto con la misma rapidez.

La máquina de Turing Determinista (DTM) es capaz de resolver problemas en tiempo polinómico y está relacionada con la clase de problemas P. Por otro lado, la máquina de Turing No Determinista (NTM) puede explorar múltiples soluciones simultáneamente hasta encontrar una correcta y está asociada con la clase de problemas NP.

A pesar de haber realizado diversos avances, no se llego a demostrar la validez del problema “P=NP”, siendo tratado por muchas personalidades como una conjetura cuya validez es desconocida y que quizás se pueda demostrar a futuro.

Una de estas personalidades que se dedico a tratar de resolver este enigma durante su vida fue Stephen Cook, quien desarrollo la teoría de la Completitud, proponiendo que dentro de los problemas NP hay problemas NP-Completos, siendo estos los problemas mas difíciles y claves dentro de los NP, ya que si se encontraba la solución en tiempo polinómico para uno de estos significaba encontrar la llave maestra para la resolución de todos los demás problemas en la clase NP.

El impacto que tendría demostrar la validez de esta teoría seria enorme en la evolución de la humanidad, no solo abarcando esta área, sino todas en general, ya que resolver un problema NP implicaría desarrollar nuevas tecnologías con algoritmos mas eficientes, lo que mejoraría su aplicación en diversas áreas como la medicina e industria.

II. Desarrollo de contenidos

A. Maquina Turing

1) Definición Conceptual

La Máquina de Turing es un modelo teórico y matemático a la vez, es decir que en realidad es una representación abstracta de un sistema, sin necesidad de realizar una experimentación física. Este sistema se basa en las leyes y propiedades matemáticas para definir diversos estados y reglas especificas que ayudan a procesar la información.

Esta compuesta por diversas partes esenciales, siendo entre estas:

1) Cabezal: Se desplaza a la izquierda o derecha a lo largo de la cinta, permitiendo realizar operaciones de lectura y escritura de símbolos en la celda actual.

2) Cinta: Es infinita y esta dividida en varias celdas, las cuales pueden contener un símbolo de un conjunto finito de símbolos.

3) Estado: Representa el estado actual en el que se encuentra la maquina, influyendo en como se procesan los símbolos en la cinta como también que decisiones se toman.

4) Función de Transición: Determina como la maquina cambia de un estado a otro, esto implica definir:
-Que símbolo escribir en la cinta.
-Cual sera el nuevo estado actual de la maquina.
-En que dirección se moverá el cabezal.

5) Alfabeto de Símbolos: Es el conjunto de simbolos que la maquina puede usar en la cinta.

6) Estado de Aceptación: Es un estado especial en el que la maquina termina su ejecución y indica que la entrada ha sido valida o aceptada.

7) Estado de Rechazo: Es otro estado especial en el que la maquina termina su ejecución y indica que la entrada ha sido rechazada.

8) Unidad de Control: Coordina el funcionamiento de la maquina de a cuerdo a las instrucciones definidas en la tabla de transición.

Todo esto fue introducido en 1936 por Alan Turing, quien produjo este modelo como resultado de querer observar los limites de lo que una maquina puede o no puede hacer, demostrando que sus capacidades no son absolutas en varios aspectos.

2) Definición Matematica

Una Maquina de Turing se define formalmente como una séxtuple, que esta conformada como:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F) \tag{1}$$

donde:

- Q: Es el conjunto finito de estados, donde cada uno representa una configuración interna de la maquina. Estos estados están denotados como $q_0, q_1, q_2, q_3, \ldots$

- Σ: Es el alfabeto, definido como conjunto finito de símbolos de entrada que la máquina puede leer de la cinta. No incluye el espacio en blanco que suele estar representado por $\square$.

- Γ: Es el alfabeto de cinta, que incluye todos los símbolos del alfabeto de entrada Σ más el símbolo blanco $\square$. Representa todos los símbolos que la máquina es capaz de escribir en la cinta.

- δ: Es la función de transición, la cual esta definida como:

$$\delta = Q \times \Gamma \to Q \times \Gamma \times \{L, R\} \tag{2}$$

  donde:

  - Q es el conjunto de estados.
  - Γ es el alfabeto de cinta.
  - L y R representan los movimientos del cabezal de lectura/escritura hacia la izquierda o la derecha.
  - $\times$ es el producto cartesiano.

  La función δ toma un par (q, γ), donde q es el estado actual y γ es el símbolo en la cinta, y retorna un triplete (q', γ', D), donde q' es el nuevo estado, γ' es el símbolo a escribir, y D es la dirección del cabezal hacia la derecha o izquierda.

- q_0: El estado inicial, es el estado con el que comienza la maquina.

- F: Es el conjunto de estados de aceptación. Es decir, cuando la maquina alcanza uno de estos estados, su proceso termina y se acepta la entrada.

3) Utilidades

La Maquina de Turing permitió realizar diversos avances en la computación, teniendo muchas utilidades, siendo las mas destacadas:

1) Ayudar a un mejor análisis de los algoritmos, ya que permite simular la ejecución de estos y visualizar paso a paso como los datos de entrada se van transformando con cada transición de estados.

2) Permitir comprender fundamentalmente como operan las computadoras, ya que al ser el modelo de donde se desarrollaron los ordenadores actuales, este nos brinda un análisis básico de cómo una computadora guarda datos, lee instrucciones, y toma decisiones paso a paso.

3) Permitir analizar si se puede resolver un problema mediante un algoritmo para todos los posibles casos de entrada.
En caso de que si sea posible, se determina al problema como decidible, caso contrario como indecidible.

4) Permitir medir la cantidad de pasos necesarios para resolver un problema en función de los datos de

entrada. Esto ayuda a calcular la complejidad temporal que tiene cada algoritmo.

5) Facilitar la demostración de que un lenguaje de programación es Turing Completo, ya que si este puede realizar un programa que simule el funcionamiento de esta maquina, entonces puede expresar cualquier operación que sea computacionalmente posible.

4) Funcionamiento

La maquina de Turing es capaz de realizar tres operaciones básicas cuando su cabezal está sobre la cinta: leer el símbolo en la celda actual, editar el símbolo por un nuevo valor y mover la cinta hacia la izquierda o derecha para leer o editar una celda adyacente. Cada símbolo leído puede estar asociado con una acción específica.
Por ejemplo, si el símbolo es un 1, la máquina puede escribir un 0 en su lugar y mover el cabezal hacia la derecha; si el símbolo es un 0, la máquina puede cambiarlo por un 1.

Como se puede observar en la fig.1, este proceso cambia los valores de 0 a 1 y viceversa, conocido como inversión, debido a que altera los valores binarios en la cinta.

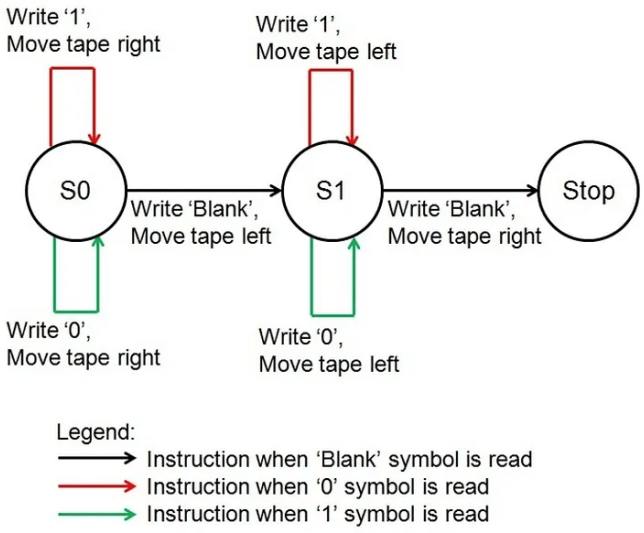

Fig. 1: Proceso de Inversión

5) Tipos de Maquinas

-Máquina de Turing Determinista (DTM)

Representa el modelo original donde comenzó todo, implementado por Alan Turing en 1936, introducido en su articulo "On Computable Numbers, with an Application to the Entscheidungsproblem".

Se lo conoce como un modelo determinista, debido a que cada uno de los resultados son predecibles y están determinados únicamente por las condiciones iniciales y el conjunto de reglas que se aplican.

En otras palabras, para cada combinación de estado y símbolo en la cinta, existe una regla especifica de transición que determina:

- El siguiente estado.
- El símbolo a escribir en la cinta.
- A que dirección gira el cabezal (izquierda o derecha).

A continuación se puede observar un ejemplo de entendimiento sobre el funcionamiento de este modelo de la Maquina de Turing.

Objetivo: Incrementar un Número Binario en 1.
Definiciones de la Máquina de Turing:

- Estado Inicial: q_0
- Estado Final (Aceptación): q_2
- Estado de Rechazo: Ninguno
- Símbolos Usados en la Entrada: {0, 1}
- Alfabeto de la Cinta: {0, 1, B}
- Símbolo de Espacio en Blanco: B
- Símbolo de movimiento del cabezal a la izquierda: L
- Símbolo de movimiento del cabezal a la derecha: R
- Símbolo de Stop, es decir que no se realizaran mas operaciones: S

Tabla de Transición:

Estado Actual	Símbolo Leído	Nuevo Símbolo	Movimiento del Cabezal	Nuevo Estado
q_0	1	1	R	q_0
q_0	0	0	R	q_0
q_0	B	B	L	q_1
q_1	1	0	L	q_1
q_1	0	1	S	q_2
q_1	B	1	S	q_2

*Ejemplo de entrada: 1011

- Inicial: **1**011 (Estado: q_0, Cabezal en el dígito 1)
- Estado q_0, Dígito leído 1: Mueve a la derecha.
  - 1**0**11 (Estado: q_0, Cabezal en dígito 0: Mueve a la derecha)
  - 10**1**1 (Estado: q_0, Cabezal en el dígito 1:Mueve a la derecha)
  - 101**1** (Estado: q_0, Cabezal en el dígito 1: Mueve a la derecha)
  - 1011**B** (Estado: q_0, Cabezal en el símbolo B)
- Estado q_0, Símbolo leído B: Pasa al estado q_1, mueve a la izquierda.
  - 1011B (Estado: q_1, Cabezal en el dígito 1)
- Estado q_1, Dígito leído 1: Cambia a 0, mueve a la izquierda.
  - 101**0**B (Estado: q_1, Cabezal en el dígito 1: Cambia a 0 y mueve a la izquierda)

- 1**0**00B (Estado: q_1, Cabezal en el dígito 0)

- Estado q_1, Dígito leído 0: Cambia a 1, pasa al estado de aceptación q_2 y finaliza la ejecución.

*Resultado final: 1100

-Maquina de Turing No Determinista(NTM)

Representa la extensión del concepto aplicado en el modelo original, realizado igualmente por Alan Turing en 1938, introducido en su articulo "Systems of Logic Based on Ordinals".

Se lo conoce como un modelo no determinista, gracias a que el resultado no se puede predecir con certeza, debido a la existencia de múltiples opciones o posibilidades a tomar.

Es decir, que para un mismo estado, se tiene la posibilidad de tomar múltiples decisiones y explorar diversas alternativas de solución hasta encontrar una que sea correcta o caso contrario que todas sean erróneas.
La búsqueda de diversas soluciones se realiza de forma simultanea, permitiendo tener una mayor eficiencia en la resolución de problemas.

A continuación se puede analizar un ejemplo de entendimiento sobre el funcionamiento de este modelo extendido de la Maquina de Turing.

Objetivo: Determinar si una cadena tiene al menos dos unos y un cero.
Definiciones de la Máquina de Turing:

- Estado Inicial: q_0
- Estado Final (Aceptación): q_f
- Estado de Rechazo: q_r
- Símbolos Usados en la Entrada: $\{0, 1\}$
- Alfabeto de la Cinta: $\{0, 1, B\}$
- Símbolo de Espacio en Blanco: B
- Símbolo de movimiento del cabezal a la izquierda: L
- Símbolo de movimiento del cabezal a la derecha: R
- Símbolo de Stop, es decir que no se realizarán más operaciones: S

Tabla de Transición:

Estado Actual	Símbolo Leído	Nuevo Símbolo	Movimiento del Cabezal	Nuevo Estado
q_0	1	1	R	q_1
q_0	0	0	R	q_5
q_0	1	1	R	q_3
q_1	1	1	R	q_2
q_1	0	0	R	q_1
q_1	B	B	S	q_r
q_2	1	1	R	q_2
q_2	0	0	S	q_f
q_2	B	B	S	q_r
q_3	1	1	R	q_3
q_3	0	0	R	q_4
q_3	B	B	S	q_r
q_4	1	1	S	q_f
q_4	0	0	R	q_4
q_4	B	B	S	q_r
q_5	0	0	R	q_5
q_5	1	1	R	q_6
q_5	B	B	S	q_r
q_6	0	0	R	q_6
q_6	1	1	S	q_f
q_6	B	B	S	q_r

*Ejemplo de entrada: 10101

Camino A (Aceptación)

- Inicial: **1**110 (Estado: q_0, Cabezal en el dígito 1)
- Estado q_0, Dígito leído 1: Pasa al estado q_1 y mueve a la derecha.
  - 1**1**10 (Estado: q_1, Cabezal en el dígito 1: Pasa al estado q_2 y se mueve a la derecha)
  - 11**1**0 (Estado: q_2, Cabezal en el dígito 1: Mueve a la derecha)
  - 111**0** (Estado: q_2, Cabezal en el dígito 0: Pasa al estado de aceptación q_f y finaliza la ejecución)

*Resultado final: Aceptado

Camino B (Rechazo)

- Inicial: **1**110 (Estado: q_0, Cabezal en el dígito 1)
- Estado q_0, Dígito leído 1: Pasa al estado q_3 y se mueve a la derecha.
  - 1**1**10 (Estado: q_3, Cabezal en el dígito 1: Mueve a la derecha)
  - 11**1**0 (Estado: q_3, Cabezal en el dígito 1: Mueve a la derecha)
  - 111**0** (Estado: q_3, Cabezal en el dígito 0: Pasa al estado q_4 y se mueve a la derecha)
  - 1110**B** (Estado: q_4, Cabezal en el símbolo B)
- Estado q_4, Símbolo leído B: Pasa al estado de rechazo q_r y finaliza la ejecución.

*Resultado final: Rechazado

-Maquina de Turing con cinta infinita a ambos lados

Representa un modelo de la Maquina de Turing donde la cinta llega a ser infinita en ambos lados, ya no solo en el lado derecho.
Esto permite eliminar la necesidad de considerar el borde de la cinta en la simulación de problemas.

-Maquina de Turing con Cinta Multiplista

Representa un modelo de la Maquina de Turing donde cada celda perteneciente a la cinta, ahora contiene subceldas. Esto permite contener múltiples símbolos, denominados como n-tuplas ordenadas.
Gracias a esta estructura, es posible representar información mas compleja en una celda, permitiendo simplificar algunos problemas al evitar la necesidad de utilizar múltiples celdas.

-Maquina de Turing Multicinta

Representa un modelo de la Maquina de Turing que posee múltiples cintas y cabezales de lectura/escritura. Cada cinta es finita en ambas direcciones y la acción a realizar depende del símbolo leído y de las reglas definidas en cada cabezal.
Esta estructura, permite procesar múltiples partes de la entrada y realizar varias operaciones complejas de forma mas eficiente.

-Maquinas de Turing Multidimensional

Representa un modelo de la Maquina de Turing que tiene una cinta, la cual se extiende infinitamente en mas de una dimensión, por ejemplo una configuración bidimensional.
Gracias a la capacidad de esta estructura de moverse en múltiples dimensiones, es posible realizar problemas que son usuales en un espacio multidimensional, como las grillas.

-Maquina de Turing Universal

Representa un modelo, capaz de simular el comportamiento de cualquier otra Maquina de Tuing. Esto es posible, gracias a una codificación adecuada, que transforma las instrucciones y datos de la maquina que se quiere simular en un formato que la Maquina de Turing Universal puede entender.

6) Problemas P

Cuando se mide la dificultad al momento de hallar la solución a un problema, como también determinar la eficiencia de una solución obtenida, existe una palabra clave denominada Polinomial y representa uno de los muchos resultados que es posible obtener.

La clase P (Tiempo Polinómico) contiene a todo ejercicio, cuya solución es fácil y pueda ser resuelto por una Maquina de Turing Determinista en un tiempo Polinomial.
En otras palabras, se considera que un problema cualquiera esta dentro de la clase P, si existe un algoritmo que lo resuelva y que su tiempo de ejecución sea de orden polinómico.

Esta clasificación surgió como parte de la teoría de la complejidad computacional en la década de 1960, cuando se decidió clasificar a los problemas en base a la cantidad de tiempo necesario para resolverlos, en función del tamaño de entrada.

Además, esta clase de ejercicios es considerada como un subconjunto de los problemas NP, ya que si un problema puede ser resuelto en tiempo polinómico, también puede ser verificado en dicho tiempo.

7) Problemas NP

Sus siglas proviene de ”tiempo no determinista polinomial”, un problema pertenece a esta clase si su solución puede ser verificada por una Maquina de Turing Determinista en tiempo polinomial, pero esta misma solución es muy difícil de encontrar. Por lo cual, se utiliza una Maquina de Turing No Determinista, con el propósito de encontrar la solución de forma eficiente, en un tiempo polinómico. Esto es posible, gracias a que esta maquina puede explorar todas las alternativas de solución simultáneamente, hasta encontrar una valida.

Sin embargo, buscar todas las alternativas de solución, implica consumir una gran cantidad de recursos, siendo inviable para las computadoras actuales, las cuales a pesar de que puedan explorar múltiples alternativas de solución, continúan siendo limitadas por la cantidad de tiempo y memoria disponibles.

En términos mas simples, encontrar la solución a un problema NP podría requerir millones de años y grandes recursos, pero que una vez encontrada dicha solución, es fácil comprobar si es correcta o no.

8) Problemas NP-Completos

La clase o categoría de este tipo de ejercicios es considerada como un subconjunto en el universo de los problemas NP.

El concepto de problema NP-Completo fue formalizado por Stephen Cook en 1971, a través de su articulo ”The Complexity of Theorem-Proving Procedures”, definiendo a este tipo de problemas como los mas difíciles dentro de los NP, pero que si se llegaba a encontrar la solución para alguno de estos, esta misma serviría como una llave maestra para la resolución de los demás problemas NP, ya que por propiedad todo problema NP puede ser reducido en tiempo polinomial

a un problema NP-Completo.

En términos mas sencillos, implicaba que si existía un algoritmo que solucionaba eficientemente alguno de los problemas NP-Completos, entonces existía un algoritmo para resolver de forma eficiente todos los demás problemas NP.

Todo este concepto realizado por Stephen Cook, seria ampliado por Richard Karp, que en 1972 a traves de su articulo "Reducibility Among Combinatorial Problems", definiría una lista de los 21 problemas NP-Completos. Entre estos problemas, los mas destacados serian:

1) Satisfacibilidad Booleana (SAT): Dado un conjunto de variables booleanas y una fórmula en lógica proposicional, se busca determinar si existe alguna asignación de valores que haga verdadera la fórmula.

2) Problema de la mochila (Knapsack Problem): Se trata de seleccionar un subconjunto de ítems, cada uno con un peso y un valor, de modo que se maximice el valor total sin exceder un peso máximo permitido.

3) Problema del Vendedor Viajero (TSP): Consiste en encontrar el recorrido más corto que permita a un vendedor viajar a través de un conjunto de ciudades, visitando cada una solo una vez y regresando al punto de partida.

4) Problema del Conjunto Independiente (Independent Set Problem): Se busca un subconjunto de vértices donde no exista una arista entre ellos.

5) Problema del Coloreado de Grafos (Graph Coloring Problem): Consiste en determinar si es posible asignar colores a los vértices de un grafo de tal forma que no haya dos vértices adyacentes con el mismo color, usando un número mínimo de colores.

B. Teoría "P=NP"

Considerado como un problema del milenio, ya que a pesar de haber sido desarrollado por un gran numero de notables personalidades, no se llego a demostrar la validez de esta conjetura.

La complejidad exponencial, representada como $O(2^n)$ o $O(n!)$, describe problemas donde el tiempo de ejecución de un algoritmo crece exponencialmente con el tamaño de la entrada. Muchos de los problemas de la clase NP son considerados como intratables para una gran cantidad de datos de entrada, ya que sus soluciones son de complejidad exponencial.

Demostrar la validez de la hipótesis "P=NP" implicaría que las clases de problemas P y NP son equivalentes, es decir, que cualquier problema cuya solución pueda ser verificada en tiempo polinómico también puede ser resuelto en dicho tiempo. Esto permitiria que muchos de los problemas que actualmente requieren tiempo exponencial para resolverse puedan ser resueltos en tiempo polinómico, es decir, en $O(n^k)$ para algún k.

Sin embargo, hasta el día de hoy, esta teoría no ha sido confirmada, lo que fortalece la idea de que la clase P podría no ser igual a la clase NP. Esto sugiere que existen problemas que no se pueden resolver de manera eficiente, aunque sus soluciones pueden ser verificadas de forma rápida mediante una Maquina de Turing Determinista.

El impacto que tendría validar esta teoría seria enorme en torno a la evolución humana, ya que permitiría desarrollar nuevos programas con algoritmos mas eficientes, mejorando su aplicación en diversas áreas del mundo como la medicina, industria y logística.
No obstante, conllevaría una gran desventaja, ya que el método de cifrado que utilizan muchos sistemas de criptografía dependen de problemas que actualmente son difíciles de resolver en un tiempo razonable.

III. Conclusiones

- La maquina de Turing, tanto en su versión determinista como no determinista, fue fundamental en el desarrollo de la complejidad computacional, permitiendo obtener una mejor comprensión sobre los limites, en relación a lo que pueden calcular las maquinas.

- La clasificación de problemas en clases P,NP,NP-Completos, proporciona una estructura para entender la dificultad de resolver distintos problemas y como se relacionan todos estos tipos de problemas entre si.

- La demostración del problema P=NP seria un logro sin precedentes, permitiendo revolucionar la eficiencia de los algoritmos y, por ende, el desarrollo tecnológico en diversos campos como la medicina, industria y logística. Sin embargo, esto también podría afectar negativamente a la seguridad de muchos sistemas criptográficos.

- Aunque se haya realizado un estudio exhaustivo sobre la validez de la conjetura P=NP a lo largo de la historia, aún no se ha podido demostrar.
  Esto mantiene abierta la posibilidad de que los problemas NP podrían ser inherentemente más difíciles de resolver que de verificar, dando a entender que algunos problemas nunca podrán resolverse eficientemente, a pesar de ser comprobables rápidamente.

References

[1] A. J. González, *NP-Completitud*, ELO320: Estructura de Datos y Algoritmos, 1er. Sem., 2002.

[2] *P vs NP: El problema más difícil de todos los tiempos*, [Online]. Available: https://platzi.com/blog/p-vs-np-problema-dificil-viajes-tiempo/. [Accessed: Aug. 25, 2024].

[3] *La complejidad del problema del milenio P vs NP y un algoritmo para su solución*, [Online]. Available: https://niboe.info/p-vs-np/. [Accessed: Aug. 25, 2024].

[4] *P vs NP: El dilema matemático que creó la complejidad computacional*, [Online]. Available: https://www.technologyreview.es/s/13775/p-vs-np-el-dilema-matematico-que-creo-la-complejidad-computacional. [Accessed: Aug. 25, 2024].

[5] *P versus NP: He ahí el dilema*, [Online]. Available: https://www.bbva.com/es/p-versus-np-he-ahi-dilema/. [Accessed: Aug. 25, 2024].

[6] *P vs NP o la utilidad de los matemáticos en juego*, [Online]. Available: https://institucional.us.es/blogimus/2022/03/pnp-o-la-utilidad-de-los-matematicos-en-juego/. [Accessed: Aug. 25, 2024].

[7] *¿Qué es una máquina de Turing y cómo funciona?*, [Online]. Available: https://formatalent.com/que-es-una-maquina-de-turing-y-como-funciona/. [Accessed: Aug. 25, 2024].

[8] *Máquina de Turing*, [Online]. Available: https://psicologiaymente.com/cultura/maquina-de-turing. [Accessed: Aug. 25, 2024].

[9] *Máquinas de Turing*, [Online]. Available: https://bootcampai.medium.com/m%C3%A1quinas-de-turing-c329ccc270f. [Accessed: Aug. 25, 2024].

[10] *Artículo en Revista Boliviana de Ciencias Sociales*, [Online]. Available: http://www.revistasbolivianas.ciencia.bo/scielo.php?lng=espid=S2415-23232016000100004script=sci$_{a}rttext$. [Accessed: Aug. 25, 2024].

[11] *Problemas de Grafos y Tratabilidad Computacional*, [Online]. Available: https://www.ic.fcen.uba.ar/ mlin/docs/PGTC/pgtc$_{c}ompl.pdf$. [Accessed: Aug. 25, 2024].

[12] *Maquina de Turing*, [Online]. Available: https://www.matesfacil.com/automatas-lenguajes/Maquina-Turing.html. [Accessed: Aug. 30, 2024].

[13] *¿Qué es una Máquina de Turing y por qué es tan importante en el mundo de la computación?*, [Online]. Available: https://www.muyinteresante.com/ciencia/63158.htmlgoogle$_{v}ignettem$. [Accessed: Aug. 30, 2024].