1 Introducción

Para conocer la complejidad de un algoritmo y compararlo con la de otros, es necesario establecer una serie de definiciones y reglas que hagan a la competencia justa. Es necesario, por ejemplo, definir en qué consiste la unidad de medida para la complejidad. Y establecer cuál es el campo sobre el cual se analiza la complejidad, que por ejemplo puede referirse a el uso de memoria o a el tiempo de ejecución.

En ésta clase se establecerán ésas definiciones y se explorará la reciente historia de su desarrollo, junto a un vistazo de su teoría.

2 Definiciones

2.1 Definiciones en lenguaje natural

Para poder introducir el concepto de la computación eficiente, podemos comenzar por establecer una serie de definiciones útiles.
Podemos comenzar por la definición de eficiencia. Una definición general sería: "La capacidad de conseguir un efecto deseado con el mínimo de recursos posibles."
En el área de las ciencias de computación, cuando nos referimos a recursos hablamos de eficiencia en el uso de memoria y eficiencia en el uso del tiempo de ejecución. "Utilizar la menor cantidad de recursos para la computación de un algoritmo."
Para el primer caso hablamos de eficiencia de memoria, y para el segundo de eficiencia temporal, dado que la cantidad de tiempo utilizado por el computador va a ser directamente proporcional a la cantidad de instrucciones a ejecutar.
Finalmente tenemos la definición de algoritmo, que puede ser definido como una receta, una secuencia de instrucciones que obedecidas proveen un resultado determinado. Un algoritmo es un conjunto de pasos determinados que se estructuran en el tiempo que responden a una lista de reglas claras y que tienen como objetivo solucionar un problema particular.

2.2 Formalización de las definiciones

Eficiencia posee definiciones que provienen del sentido común, la intuición.
Algoritmo posee definiciones formalizadas por matemáticos durante el siglo XX.
Hasta ahora hemos explorado las definiciones más intuitivas de ambos, pero para poder incursionar en el cálculo de complejidad computacional va a ser necesario definir algoritmo con mayor especificidad.

Sobre la formalización de la definición de algoritmo

Ésta sección va a dar un breve repaso histórico donde se van a mencionar los mayores participantes de su formalización. La historia, que termina con la equivalencia entre las definiciones formales de algoritmo, permite justificar el uso de las Máquinas de Turing como estándar para el desarrollo de cualquier algoritmo existente.

Historia. Desde Entscheidungsproblem hasta la Tesis Church-Turing

1900
Hilbert publicó en el Congreso Internacional de Matemáticas de París una lista de veintitrés desafíos matemáticos como desafíos propuestos para el siglo. Eran veinticuatro inicialmente, pero uno de ellos no fue presentado en el congreso, aunque se conserva en los manuscritos de Hilbert. El desafío diez, llamado Entscheidungsproblem, era el que pedía un proceso acorde al cual, dado un polinomio, pueda determinar si posee una raíz entera. O escrito con más precisión, encontrar un algoritmo que determine si una ecuación diofántica polinómica dada con coeficientes enteros tiene solución entera.
El problema no era resoluble. Pero demostrar su imposibilidad necesitaba una definición formal de algoritmo. Las definiciones intuitivas no bastaban.

1936
Alonzo Church y Alan Turing demuestran formalmente el concepto de algoritmo. El primero mediante la invención de el lambda-calculus, el segundo mediante la invención de la máquina de Turing. Ambas definiciones son consideradas equivalentes. Kleene también proveyó una definición formal de algoritmo utilizando funciones recursivas. Ésta demostración también se la considera equivalente a las anteriores.

1976
Yuri Matiyasevich, basado en el trabajo de Martin Davis, Hilary Putnam, y Julia Robinson demostró que no existe un algoritmo que cumpla los requisitos de Hilbert.

Tesis Church-Turing

La tesis de Church, o de Church-Turing, o aquella que generalmente se expresa mediante cualquier otra combinación de nombres de los protagonistas de ese período – excepto, tal vez, Gödel- es aquella propone la equivalencia entre las definiciones de algoritmos provistas por los mismos.
Cuando decimos que las definiciones de Church, Turing y Kleene se las considera equivalentes, es porque no ha sido posible demostrar su inequivalencia. En la comunidad científica se considera a la proposición de su equivalencia como la Tesis Church-Turing, donde múltiples trabajos exponen cómo lo que es posible expresar mediante los lenguajes propuestos se traduce a los otros.
No es un teorema matemático, es una afirmación formalmente indemostrable, una hipótesis que, no obstante, tiene una aceptación prácticamente universal.
Asumiendo que la Tesis Church-Turing es verdadera, todo algoritmo puede ser expresado entonces como una máquina de Turing.

"Toda tarea ejecutable de manera efectiva es computable."
"Cualquier forma de computar el Universo puede simularse con una máquina de Turing"

Teoremas de incompletitud de Gödel

El décimo desafío de Hilbert planteó si las matemáticas eran completas, si cada proposición podía ser demostrada o refutada dentro de las matemáticas.
Gödel teorizó sobre la incompletitud de los sistemas axiomáticos, pero, tras varios intentos, fue Alan Turing quien apoyó esta teoría con la creación de su máquina, minuciosamente ideada para que pudiera interpretar cualquier algoritmo, y que sin embargo tenía problemas fuera de su alcance. Concretizó la definición de algoritmo y luego desarrolló un ejemplo concreto no computable, llamado *The Halting Problem*.

Máquina de Turing como estándar

Dado que todo lo computable puede ser expresado como una máquina de Turing, estudiar eficiencia sobre las máquinas de Turing es equivalente a estudiar eficiencia sobre cualquier modelo computacional.
Ésta es la razón para el estudio de las máquinas de Turing durante el estudio de algoritmos, y en éste caso, para el estudio de la complejidad de los mismos.

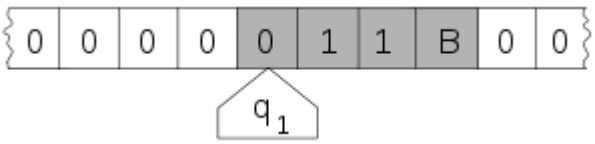
3 Máquina de Turing

"... una ilimitada capacidad de memoria obtenida en la forma de una cinta infinita marcada con cuadrados, en cada uno de los cuales podría imprimirse un símbolo. En cualquier momento hay un símbolo en la máquina; llamado el símbolo leído. La máquina puede alterar el símbolo leído y su comportamiento está en parte determinado por ese símbolo, pero los símbolos en otros lugares de la cinta no afectan el comportamiento de la máquina. Sin embargo, la cinta se puede mover hacia adelante y hacia atrás a través de la máquina, siendo esto una de las operaciones elementales de la máquina. Por lo tanto cualquier símbolo en la cinta puede tener finalmente una oportunidad."

Turing (1948, p. 61.)

El anterior texto es una cita traducida al español de un ensayo de Turing llamado "Máquinas Inteligentes", realizado en 1948.

Descripción Informal



Aquí se muestra el estado interno (q1) dentro del cabezal, y la ilustración describe la cinta como siendo infinita y llenada previamente con '0', que representa vacío. El estado completo del sistema (su configuración) consiste del estado interno(q1), el contenido de las casillas no vacías, ("11B"), y la posición del cabezal relativa a ésas casillas, incluyendo vacíos, i.e.: "011B".

La máquina de Turing modela matemáticamente a una máquina que opera mecánicamente sobre una cinta. En esta cinta hay símbolos que la máquina puede leer y escribir, uno a la vez, usando un cabezal lector/escritor de cinta. La operación está completamente determinada por un conjunto finito de instrucciones elementales como

"en el estado 42, si el símbolo visto es 0, escribe un 1; Si el símbolo visto es 1, cambia al estado 17; en el estado 17, si el símbolo visto es 0, escribe un 1 y cambia al estado 6; etc".

En su artículo ("Sobre números computables con una aplicación al Entscheidungsproblem"), Turing no imagina un mecanismo, sino una persona a la que él llama la "computadora", quien ejecuta servilmente estas reglas mecánicas deterministas (o como Turing pone, "de una manera desganada").

Más precisamente, una máquina de Turing consta de:

1. Una **cinta** que se divide en celdas, una al lado de la otra. Cada celda contiene un símbolo de algún alfabeto finito. El alfabeto contiene un símbolo especial llamado blanco (aquí escrito como 'B') y uno o más símbolos adicionales. La cinta se supone que es arbitrariamente extensible hacia la izquierda y hacia la derecha, es decir, la máquina de Turing siempre es suministrada con tanta cinta como necesite para su computación. Las celdas que no se hayan escrito previamente se asumen que están rellenas con el símbolo blanco. En algunos modelos la cinta tiene un extremo izquierdo marcado con un símbolo especial; la cinta se extiende o es indefinidamente extensible hacia la derecha.

2. Un **cabezal** que puede leer y escribir símbolos en la cinta y mover la cinta a la izquierda y a la derecha una (y sólo una) celda a la vez. En algunos modelos el cabezal se mueve y la cinta es estacionaria.

3. Un **registro de estado** que almacena el estado de la máquina de Turing, uno de los estados finitos. Hay un estado inicial especial con el que el registro de estado se inicia. Turing escribe que estos estados reemplazan el "estado de la mente" en que ordinariamente estaría una persona realizando cálculos.

4. Una **tabla** finita de instrucciones (llamada ocasionalmente como **tabla de acción** o **función de transición**). Las instrucciones son usualmente 5-tuplas: qiaj→qi1aj1dk, que, dado el estado (qi) en que la máquina se encuentra actualmente y el símbolo (aj) que se está leyendo en la cinta (el símbolo actualmente debajo del cabezal) le indica a la máquina hacer lo siguiente en secuencia:

    - Borra o escribe un símbolo (reemplazando aj con aj1), y *entonces*
    - Mueve el cabezal (que es descrito por dk y puede tener los valores: 'L' para un paso a la izquierda, o 'R' para un paso a la derecha, o 'N' para permanecer en el mismo lugar) y luego
    - Asume el mismo o un nuevo estado como prescrito (ve al estado qi1).

Notar que cada parte de la máquina — su estado y colecciones de símbolos — y sus acciones — imprimir, borrar, movimiento de la cinta — es finito, discreto y distinguible; es la cantidad potencialmente ilimitada de cinta lo que le da una cantidad ilimitada de espacio de almacenamiento.

Descripción Formal

Una máquina de Turing es un modelo computacional que realiza una lectura/escritura de manera automática sobre una entrada llamada cinta, generando una salida en esta misma.

Este modelo está formado por un alfabeto de entrada y uno de salida, un símbolo especial llamado blanco (normalmente b, $\triangleright$, o $\emptyset$), un conjunto de estados finitos y un conjunto de transiciones entre dichos estados.

Su funcionamiento se basa en una función de transición, que recibe un estado inicial y una cadena de caracteres (la cinta, la cual puede ser infinita) pertenecientes al alfabeto de entrada. La máquina va leyendo una celda de la cinta en cada paso, borrando el símbolo en el que se encuentra posicionado su cabezal y escribiendo un nuevo símbolo perteneciente al alfabeto de salida, para luego desplazar el cabezal a la izquierda o a la derecha (solo una celda a la vez).

Esto se repite según se indique en la función de transición, para finalmente detenerse en un estado final o de aceptación, representando así la salida.

Una máquina de Turing con una sola cinta puede definirse como una 7-tupla

$\mathcal{M}$= (Q, Σ, Γ,s, b, F, δ)

donde:

Q es un conjunto finito de estados

Σ es un conjunto finito de símbolos distinto de blanco, denominado alfabeto de entrada.

Γ es un conjunto finito de símbolos de cinta, denominado alfabeto de cinta ($\Sigma \subseteq \Gamma$).

s$\in$ Q es el estado inicial.

b$\in \Gamma$ es un símbolo denominado blanco,
es el único símbolo que se puede repetir un número infinito de veces.

F$\subseteq$Q es el conjunto de estados finales de aceptación.

δ :Q$\times\Gamma \rightarrow$ Q$\times\Gamma\times$ {L,R}
es una función parcial denominada función de transición,
siendo L es un movimiento a la izquierda y R es el movimiento a la derecha.

4 Ejemplo concreto de una Máquina de Turing

Una máquina de Turing sencilla podría ser aquella que determinase, dada una secuencia de letras, si es posible invertirla de manera tal que conserve su valor original. Dicho de otra manera, que determine si una frase es un palíndromo.

A ésta máquina de Turing la llamaremos $\mathcal{M}_{\text{Pal}}$.

Suponiendo que tenemos dos cintas, usaremos la primera para ingresar el input y la segunda para procesarlo.

Una tercer cinta será usada para escribir el output del algoritmo computado.

1. Mover el cabezal en la primer cinta hasta alcanzar el final de datos ingresados.
2. Comenzar a leer hacia atrás, hasta el comienzo de los datos ingresados.
3. En cada paso hacia la izquierda, escribir los símbolos en la segunda cinta, hasta llegar al comienzo de los datos ingresados. Ésto efectivamente va a escribir los datos en la segunda cinta de manera invertida.
4. Mover ambos cabezales hasta el comienzo.
5. Comenzar a leer hacia adelante, con ambos cabezales.
6. En cada paso hacia la derecha, comparar los valores de cada cabezal:
   - Si los símbolos son diferentes, el input no es un palíndromo, y se escribe se termina la ejecución, escribiendo en la cinta de output un valor que represente a el símbolo halt-reject. Usualmente se lo representa con un cero.
   - Si los símbolos son iguales, el input puede ser un palíndromo, y se continúa avanzando.
     - Si se alcanza el fin de las cintas, se termina la ejecución, escribiendo en la cinta de output un valor que represente a el símbolo halt-accept Usualmente se lo representa con un uno.

maybe add a graph representation of $\mathcal{M}_{\text{Pal}}$ as a state machine *here*