Máquinas de Turing

¿Por qué un modelo de cómputo más potente?

Ya sabemos que existen lenguajes que no pueden ser reconocidos con memoria finita. Esto implica que hay problemas que no pueden ser resueltos con memoria finita.

¿Cómo extendemos el modelo DFA para tener memoria infinita? No podemos poner una cantidad infinita de estados (¿por qué?)

La solución será permitir que el autómata **escriba** en una *memoria potencial infinita*. Para ello, además, habrá que permitir al autómata que se desplace de forma arbitraria por la memoria.

¿Cómo formalizamos esto entonces? ¿Cuál es el mecanismo más simple que tiene memoria arbitrariamente grande y manipulable sin restricción?

Tendremos los siguientes componentes:

- Una cinta unidimensional potencialmente infinita en ambas direcciones
- Un cabezal ubicado en una posición de la cinta, que puede desplazarse a la derecha o izquierda, además de escribir en la cinta.
- La cadena de entrada vendrá también en la cinta.
- Una secuencia de instrucciones en forma de estados con transiciones que indican a dónde moverse y qué poner en la cinta.

Formalmente, una máquina de Turing es una tupla $M =< Q, \Sigma, \Gamma, \delta, q_0, B, F >$, donde:

- Q: es el conjunto de estados
- Σ: es el alfabeto de entrada
- Γ: es el alfabeto de cinta ($\Sigma \subset \Gamma$)
- δ: es la función de transición, de la forma $\delta : Q \times \Sigma \to Q \times \Gamma \times \{\Leftarrow, \Rightarrow, \square\}$
- q_0: es el estado inicial
- B: es el símbolo nulo ($B \in \Gamma - \Sigma$)
- F: es el conjunto de estados finales

Cómo funciona una TM

Primero definamos una notación para describir un estado instantáneo de una TM. Es una cadena de la forma: $X_1X_2 \dots X_{i-1}qX_iX_{i+1} \dots X_n$ donde:

- $X_i \in \Gamma$ es el estado de la cinta entre el primer y el último símbolo B (o sea, a la izquierda y derecha solo quedan Bs)
- $q \in Q$ representa el estado actual, así como la posición del cabezal, que está justo sobre el símbolo X_i.

Con esta notación podemos describir formalmente cómo funciona una TM, describiendo cuando una secuencia de descripciones instantáneas es válida (queda orientado).

Más importante aún, podemos definir formalmente el lenguaje reconocido por una máquina de Turing.

El lenguaje de una TM

Intuitivamente, podemos decir que una TM acepta una cadena ω si, de la configuración $q_0w_1 \dots w_n$ se llega eventualmente a cualquier configuración $X_1 \dots q_f \dots X_n$ tal que $q_f \in F$.

O sea, la TM acepta una cadena si en alguna cantidad finita de pasos entra a un estado final.

Una vez que una TM entra a un estado final, no tiene sentido seguir computando, por lo que podemos asumir que una TM para en todo estado final, o sea que $\delta(q_f, X)$ no está definido. Igualmente podemos para todo estado no final q_j, donde quiera que $\delta(q_j, X)$ no esté definido, poner una transición a un nuevo estado q_{loop} que haga un ciclo infinito.

Por tanto, podemos definir un criterio alternativo (y equivalente) de aceptación que es decir una *TM acepta una cadena si se detiene en algún momento*.

De esta definición saldrá el problema más importante de la teoría de la computabilidad, el famoso *halting problem*.

> ***NOTA:*** *Con esta transformación podemos hacer que una TM que acepta cualquier lenguaje por el criterio de estado final, nos de una TM que acepta el mismo lenguaje por el criterio de parada (y que si no aceptaba antes, pues ahora no pare en esas cadenas).*
>
> *Lo que no podemos garantizar es, para toda máquina de Turing que acepta por criterio de parada, siempre convertirla a una TM que para en toda entrada, pero en estados no finales para las cadenas que no aceptaba. Esto lo veremos en detalle en la siguiente conferencia.*

Máquinas de Turing como computadoras

Además de vez de ver una Máquina de Turing como un reconocedor de cadenas, podemos ver la máquina como un mecanismo de cómputo que, una vez termina (si termina), nos deja en la cinta la respuesta a la entrada correspondiente.

Por tanto, una TM puede hacer dos cosas:

- Validar la entrada (acepta o no)
- Computar una salida

Depende de cómo interpretemos la TM nos interesará lo que queda en la cinta o no.