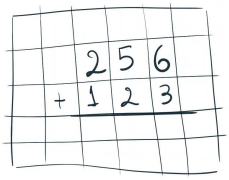
Máquinas de Turing

0. La idea de Turing (1935).

- La pregunta es: ¿Qué es computar?
- La idea de Turing es responderla considerando lo que hace una persona que se encuentra realizando un cálculo.



Pensemos entonces en una hoja cuadriculada, donde hay escritos ciertos símbolos como por ejemplo los de la figura.

En este ejemplo consideramos la realización de una suma. Entonces uno comienza por observar un cierto casillero (digamos el que contiene el "6").
En cada momento, uno tiene en mente cierto estado de la computación (en este caso, estamos considerando una cifra del primer número a sumar) y ello, junto con la información del casillero observado determina la acción subsecuente (que en este caso es pasar al casillero inmediatamente inferior, recordando el ya leído para ejecutar la suma, escribirla, etc.)
Entonces se tiene:

- casilleros (lugares) con símbolos (datos)
- estados de la computación
- acciones

de tal modo que:

- en cada momento, uno se encuentra en un cierto estado de la computación y observando un casillero.
- la pareja (estado, dato observado) determina una acción a realizar y un cambio de estado de la computación.

Turing formalizar esta idea definiendo una "máquina de computar" (máquina de Turing) como:

- Una cinta doblemente infinita dividida en casilleros

(Esta representa la hoja cuadriculada pero en un formato lineal. Es infinita para hacer posible considerar computaciones con datos de cualquier tamaño)

– Uno de los casilleros es el "actualmente observado"(uno podría considerar que sobre un casillero de la cinta se ubica una 'cabeza lectora':

– La máquina se describe como una tabla de la siguiente forma:

Estado de la computación	Símbolo leído		Acción	Nuevo Estado

⋮

donde cada combinación (estado de la computación, símbolo leído) determina una pareja (acción, nuevo estado).

– Los símbolos a usar conforman un conjunto finito, llamado el alfabeto de la máquina. A los símbolos del alfabeto, se agrega uno especial, llamado "blanco" y denotado por "#" que significa "ausencia de dato". Más específicamente, la cinta contendrá en cada momento solamente una cantidad finita de símbolos del alfabeto ("datos reales") y el resto de los casilleros (por lo tanto una cantidad infinita a ambos lados de la cinta) estarán ocupados por #.

– la cantidad de estados es finita. Uno de ellos será distinguido como el "estado inicial" en el cual la máquina comienza su actividad. Y se usará "h" (o a veces "ς") para denotar el estado final en el que la máquina se detiene.

– Las acciones posibles son:

  – mover la cabeza lectora a la izquierda
  – moverla a la derecha
  – sobreescribir el casillero actual con un símbolo (del alfabeto o #).

1. Primeros ejemplos: La siguiente máquina (en forma de tabla) efectúa la negación booleana de la siguiente manera: Comienza leyendo el símbolo que representa un booleano ("T" o "F") y lo sobreescribe con su opuesto:

Estado	Símbolo		Acción	Nuevo Estado
inicial	T		F	h
inicial	F		T	h

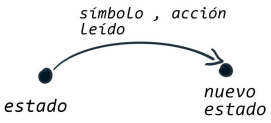
Aquí usamos la convención de que "escribir un símbolo σ" se representa simplemente por el propio σ. Moverse a la izquierda será simplemente representado por l(`left`) y moverse a la derecha, mediante r(`right`).

2. Analogía: La cinta es la memoria, en tanto la tabla es el código. Las máquinas de Turing son modelos con memoria (imperativos) que actúan tomando ésta (osea la cinta) en un cierto estado inicial y modificándola sucesivamente a través de las acciones elementales para producir un estado final que refleje la computación deseada.

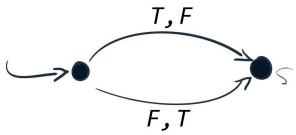
3. Diagrama de Estado

Las tablas se pueden representar mediante grafos cuyos nodos son los estados de la computación y las transiciones (aristas) van etiquetadas con el símbolo leído y la acción a realizar:

El estado inicial se marca con la flecha exterior.



El ejemplo precedente queda simplemente:

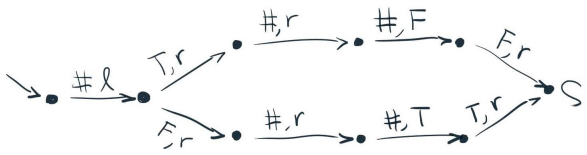


Uno debe especificar el problema a realizar por una máquina de Turing determinando entonces cuál es la configuración de la cinta al comienzo y cuál es la correspondiente configuración final. Por ejemplo, otra especificación de la negación sería:

Se tiene un booleano **b** en la cinta, seguido de un #. La cabeza se encuentra sobre éste.

Se niega el booleano escribiendo $\bar{b}$ pero en el casillero a la derecha del blanco donde la cabeza se encuentra originalmente, y se deja la cabeza lectora sobre un # a la derecha de la salida producida. (De esta manera se preserva el input original)

Una máquina (en forma de diagrama de estados) que resuelve esto es:

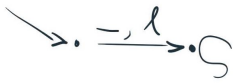


Los diagramas de estados tienden a crecer en complejidad. Estudiaremos una notación un tanto más abstracta para programar máquinas de Turing.

4. Máquinas básicas y sus combinaciones

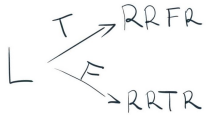
La idea es utilizar un número reducido de máquinas básicas y disponer de mecanismos de combinación.

- Por ejemplo, la máquina L es la siguiente:



  Aquí el símbolo _ (guión bajo, underscore) representa "cualquier símbolo". La máquina L simplemente se mueve un lugar a la izquierda y se detiene. (En realidad, como tabla esta máquina se tendría que escribir con una entrada para cada símbolo del alfabeto, más el # para indicar que en todos esos casos se realiza simplemente el movimiento a la izquierda. Todo ello es resumido mediante el empleo de _. Además ahora resumimos la máquina completa mediante el empleo de L.)

- Igualmente existe la máquina R
- Y ahora podemos considerar la composición secuencial de dos máquinas. Esto se logra conectando (i.e. haciendo coincidir) el estado final de la primera con el inicial de la segunda. Así LR es la composición secuencial de las dos máquinas precedentes (Una forma de no hacer nada...)
- Así también existe, para cada símbolo σ la máquina $\boldsymbol{\sigma}$ que simplemente escribe este símbolo en el casillero actual.
- La composición condicional de máquinas corresponde a la propiedad de ejecutar distintas transiciones según el símbolo que esté siendo leído al terminar la ejecución de una máquina. Por ejemplo, la siguiente máquina realiza la negación según la última de las especificaciones dadas (sin sobreescribir el input):



Se puede apreciar la simplificación respecto al correspondiente diagrama de estados)

– Otras máquinas básicas son las de búsqueda:

Esta máquina busca el primer símbolo σ que se encuentra estrictamente a la izquierda del actual y queda posicionado sobre él.

Observaciones:

1. Nótese que lo primero que hace es moverse a la izquierda sin importar el símbolo leído. Por ello decimos que busca el primer σ estrictamente a la izquierda del actual (si éste fuera un σ sería ignorado por L_σ)
2. $\overline{\sigma}$ es una notación para significar: cualquier símbolo diferente de σ. La máquina persiste en moverse a la izquierda mientras no encuentra σ. Cuando esto finalmente ocurre (si es que alguna vez ocurre) entonces lo sobreescribe por sí mismo y queda posicionado en ese lugar.
3. Esta máquina puede colgarse (si no hay ningún σ estrictamente a la izquierda del casillero actual).

– Simplemente se construye R_σ que busca el primer σ estrictamente a la derecha del casillero actual.

5. Otros ejemplos: Los números naturales se acostumbran a representar en notación unaria (secuencias de "palitos" $\langle$"|"$\rangle$, en forma similar a la notación con `0` y `S` usada en los otros modelos de computación).

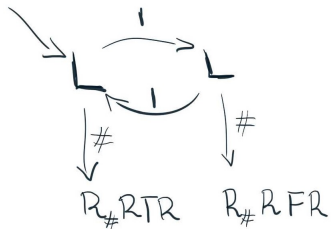
- Consideremos este problema: Se recibe un número en unario :

con la cabeza en el primer # a la derecha del número.

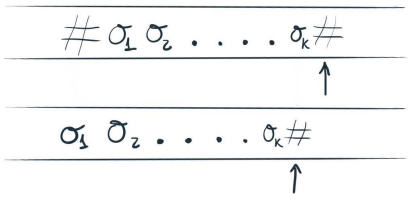
Se retorna:

el número seguido de un booleano que indica si aquél era par o no.

La idea es recorrer el número alternando sucesivamente entre dos estados. Uno corresponde a haber leído un número para de "|" y el otro a haber leído un número impar de ellos. Inicialmente, se ha leído un número para (0):

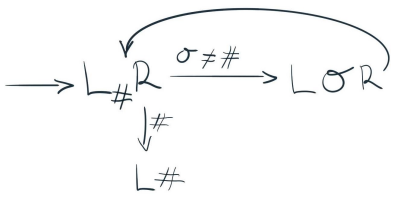


- "shift left" es una máquina que mueve una palabra entera un casillero a la izquierda. Se entiende por palabra una sucesión de símbolos no blancos.



En la figura se pueden ver la entrada esperada y la salida, además se puede apreciar (debido a que están alineadas) el movimiento de la palabra recibida.

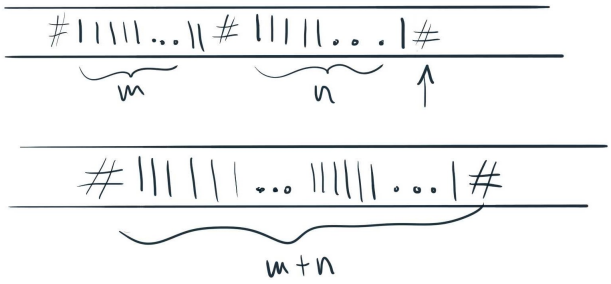
La idea es recorrer la palabra de izquierda a derecha copiando cada símbolo un lugar a la izquierda. Inicialmente nos posicionamos en el blanco de la izquierda de la palabra usando $L_{\#}$. Luego, durante el ciclo de recorrida, tendremos a nuestra izquierda la posición de la palabra que ya ha sido trasladada. Procedemos moviéndonos a la derecha y detectando qué símbolo debe ser copiado un lugar a la izquierda.



Algunos comentarios para esta solución son pertinentes:

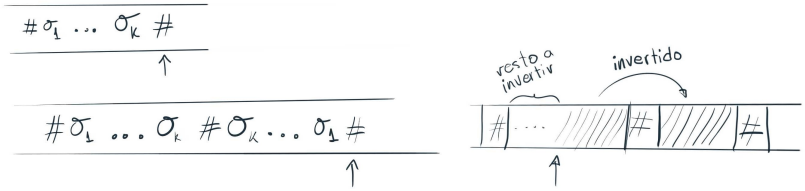
1. Es necesario borrar el último símbolo que queda repetido al final. (eso es puede ver en el último paso de la máquina)
2. En la sección derecha del ciclo, luego de haber copiado el símbolo un lugar a la izquierda, nos movemos a la derecha para continuar teniendo a la izquierda la porción ya copiada de la palabra.

- La suma de naturales, con la especificación siguiente:

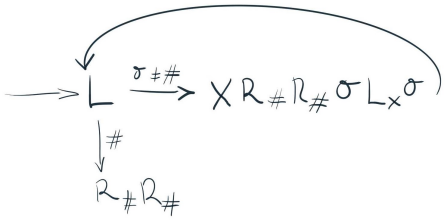


Notar que la solución de esta máquina se puede realizar simplemente con Shift left.

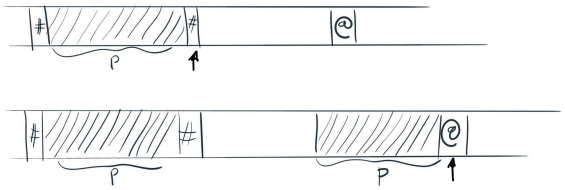
- Reversa de una palabra:



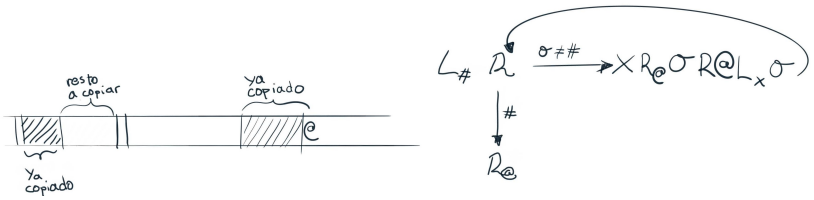
En cada momento tendremos una porción de la palabra ya invertida. Estaremos posicionados el final de esa porción. Nos movemos a la izquierda, detectamos el símbolo σ a copiar y lo sobreescribimos con "X" para poder volver aquí luego. (Este símbolo X es agregado al alfabeto)



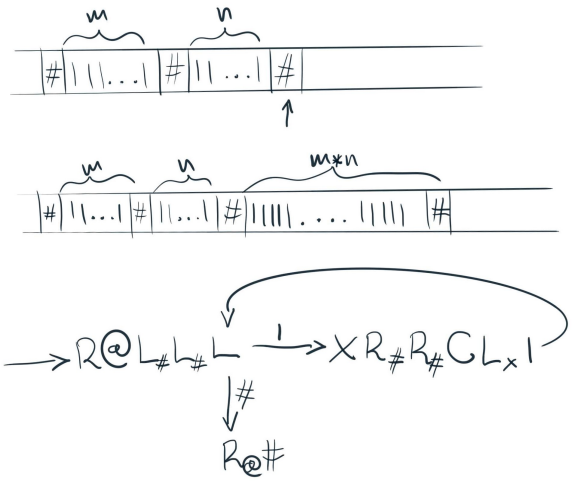
- Copia “a distancia”



Una aclaración importante es que el símbolo @ no se utiliza en otro lugar de la cinta más que el indicado.



- Producto de dos naturales



Para resolver este problema utilizamos la máquina de copia a distancia denominándola *C*.

La idea es ir copiando n sucesivamente a la derecha, una vez por cada unidad de m. Para eso se usa @ a fin de marcar el lugar desde donde debe hacerse la copia.